




Toward Brain Computer Interface: Deep Generative Models for Brain Reading

Mengmi Zhang

Mengmi.Zhang@childrens.Harvard.edu

CBMM Summer Course 2020

DeepFake in Zoom

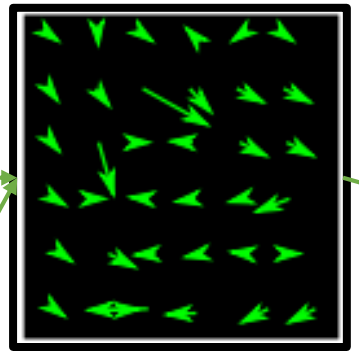
Siarohin, et al, First Order Motion Model for Image Animation, NIPS, 2019
https://github.com/alievk/avatarify



Source



Driving Frame



Optical Flow

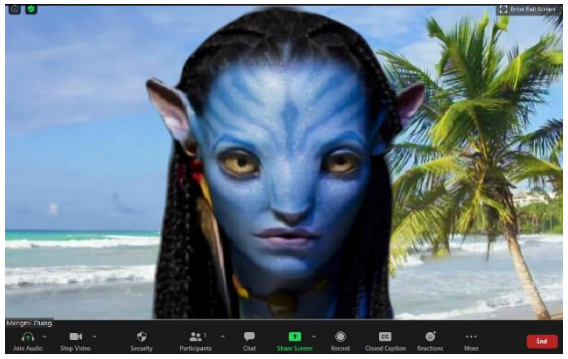


Occlusion Map

Encoder

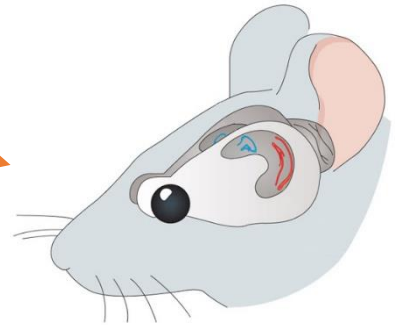
Latent Representation

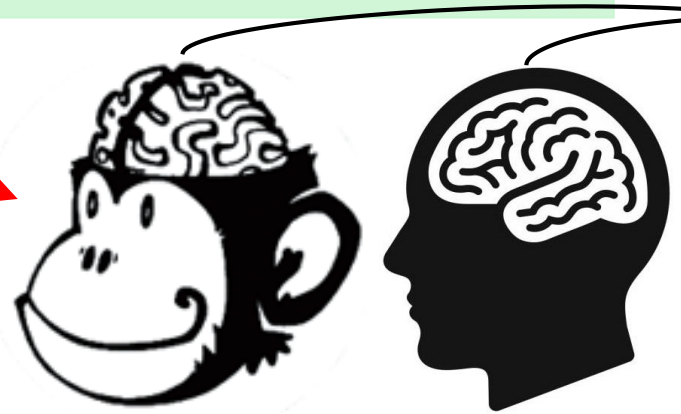
Generator



Deep Learning in Machine Learning

Supervised Learning	Unsupervised Learning	Reinforcement Learning
• Multi Layer Perceptron • Convolutional Neural Net • Recurrent Neural Net	• Variational Autoencoder • Generative Adversarial Network • Restricted Boltzmann Machine	• Deep Q Network • Actor Critic Networks • Distributional Reinforcement Learning







Seen

Reconstructed

Image credits: http://www.visembryo.com/story4341.html; https://en.wikipedia.org/wiki/MonkeyBrain_Books; https://www.istockphoto.com/illustrations/human-brain?mediatype=illustration&phrase=human%20brain&sort=mostpopular

Outline

Deep Convolutional Generative Adversarial Network (DCGAN)

BigBiGAN

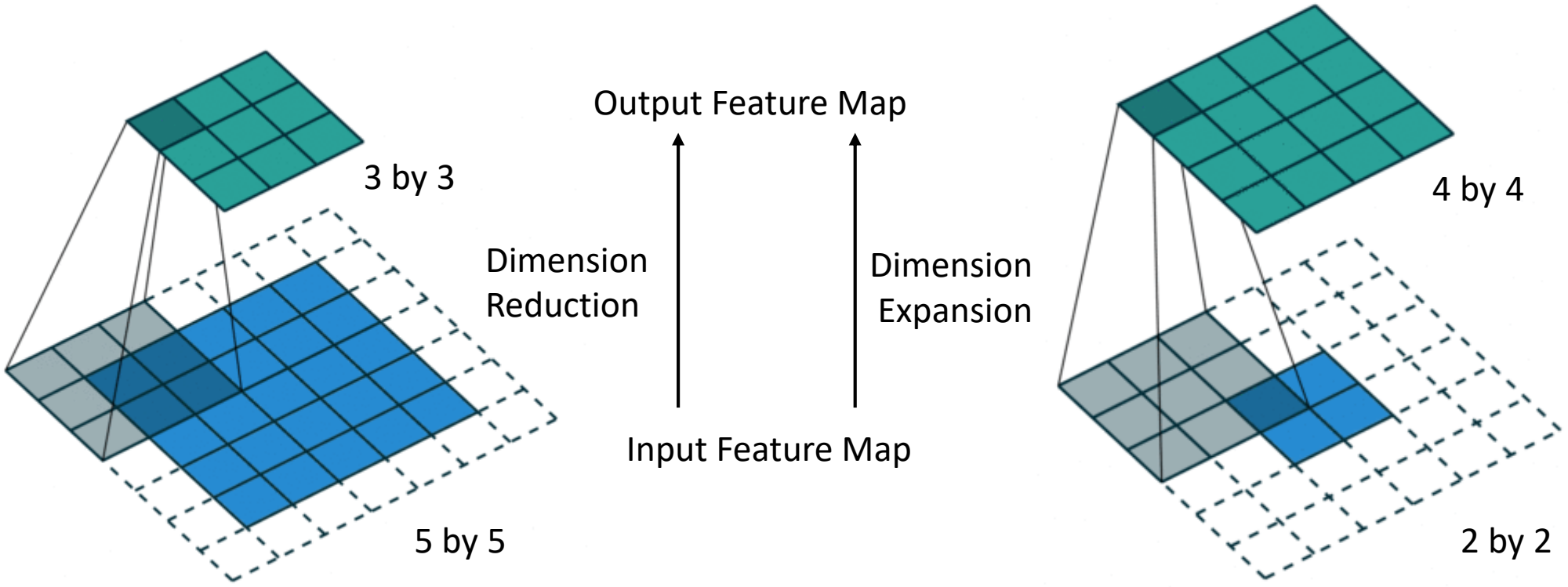
Tricks for more realistic image construction using GANs

Image reconstruction from brain signals

Evaluation of brain readers

Towards Brain Computer Interface (BCI)

Deconvolution



Pytorch: torch.nn.ConvTranspose2d(*in_channels, out_channels, kernel_size, stride, padding)*

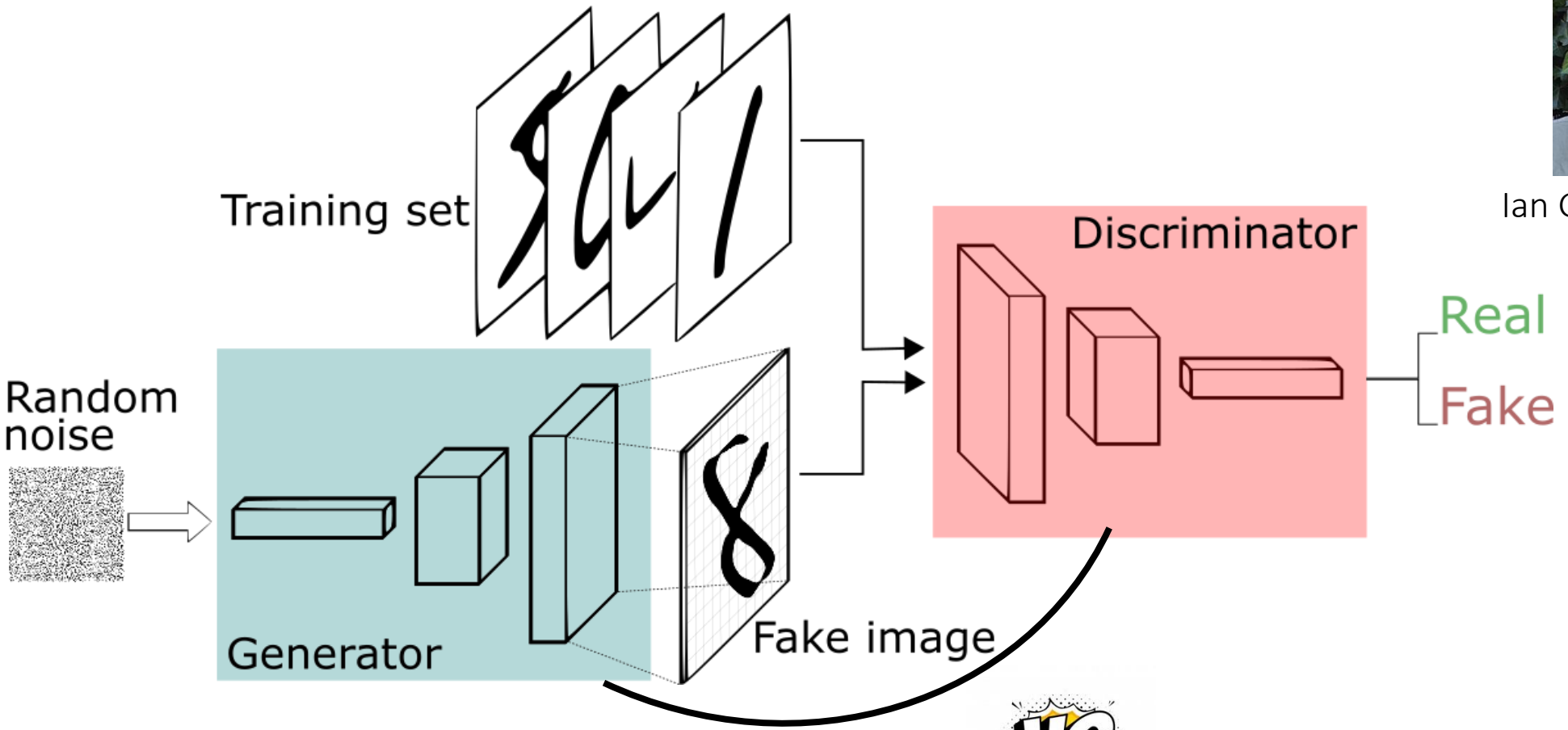
torch.nn.ConvTranspose2d(in_channels=*1, out_channels=1, kenel_size=3, stride=0, padding=2)*

Image credits: https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

Generative Adversarial Networks



Ian Goodfellow, 2014



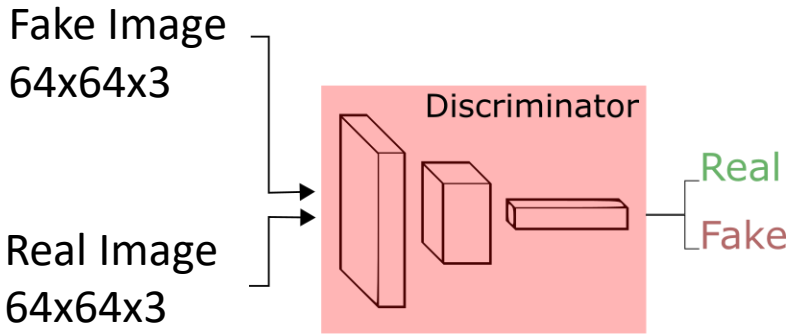
Generator VS Discriminator

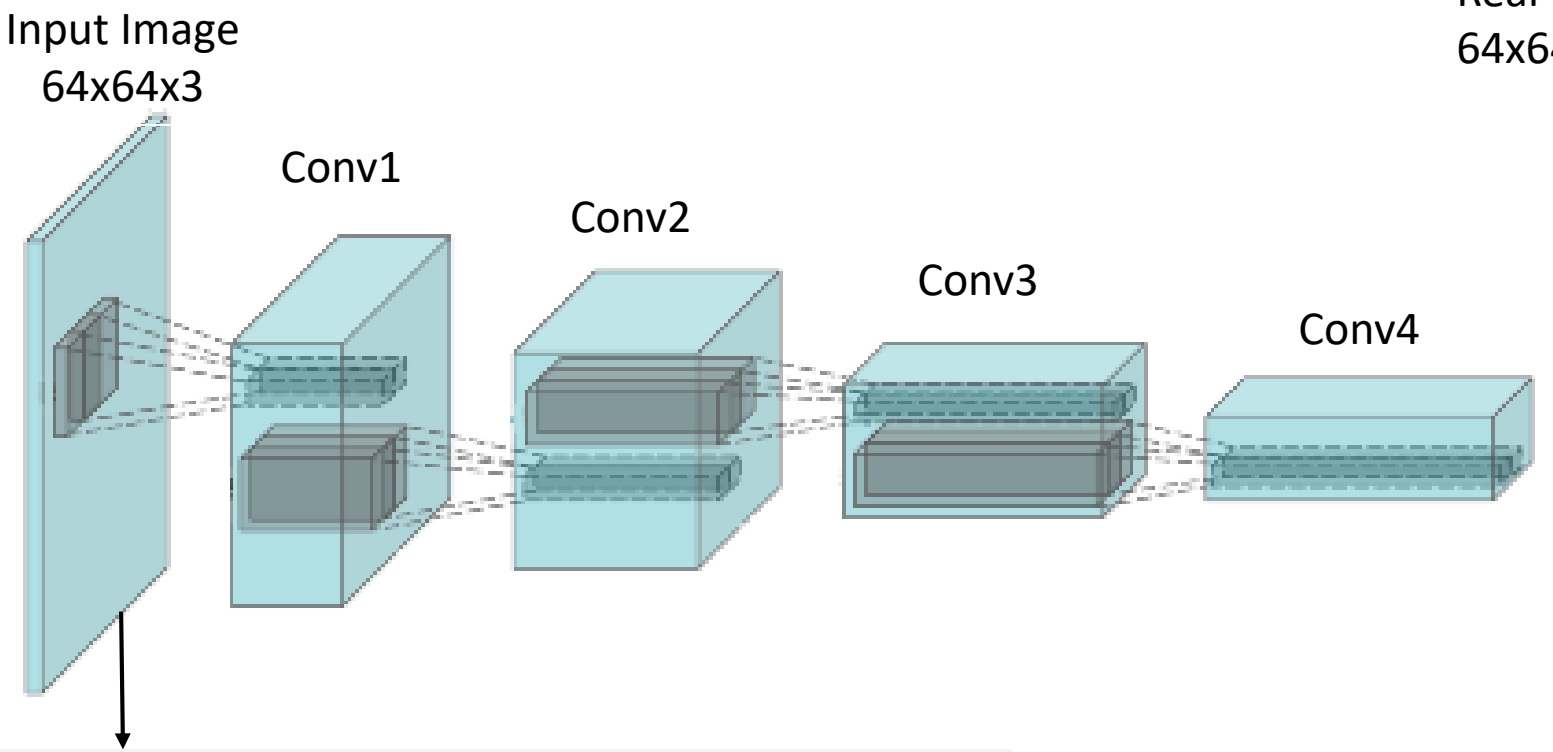
Min-max Game: minimize the possible loss for a worst scenario

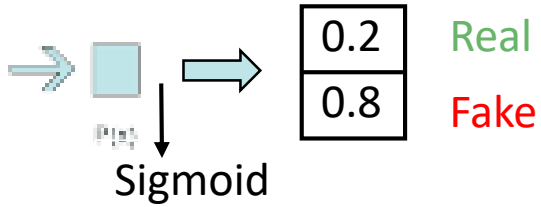
Image credit: https://sthalles.github.io/intro-to-gans/

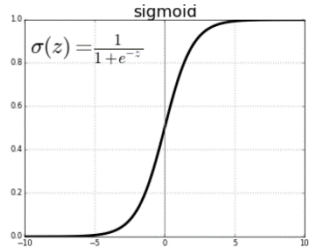
Discriminator

Deep Convolutional GAN (DCGAN)










```
# Create the dataset
dataset = dset.ImageFolder(root=dataroot,
                           transform=transforms.Compose([
                               transforms.Resize(image_size),
                               transforms.CenterCrop(image_size),
                               transforms.ToTensor(),
                               transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5)),
                           ]))
```

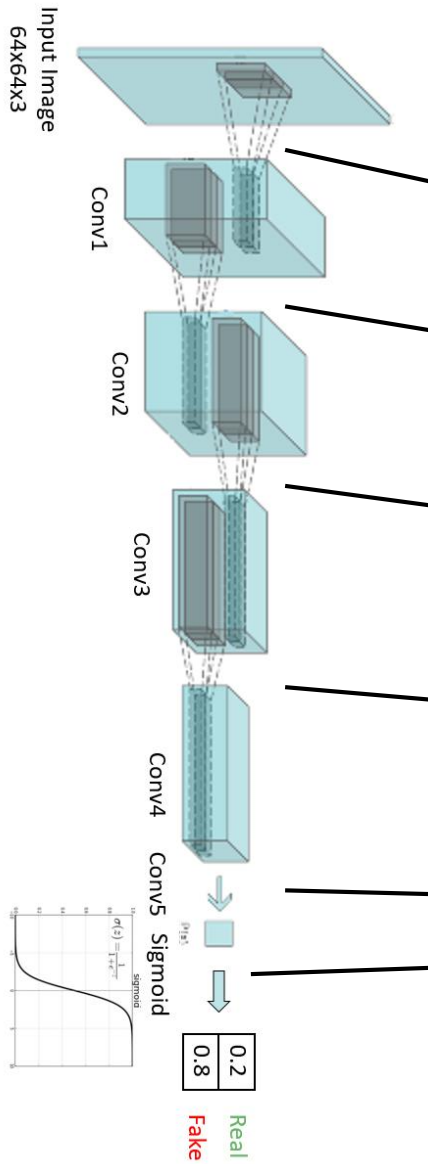
Image augmentation:
scaling, rotation, random cropping, etc

Normalize to [0,1]

Normalize image pixel value from [0,1] to [-1,1]

Image credit: https://gluon.mxnet.io/chapter14_generative-adversarial-networks/dcgan.html

Discriminator – pytorch implementation



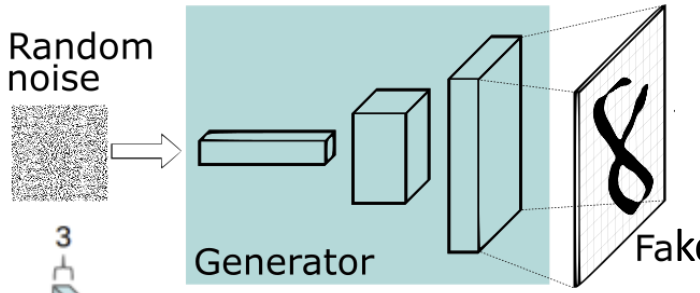

```python
class Discriminator(nn.Module):
    def __init__(self, ngpu):
        super(Discriminator, self).__init__()
        self.ngpu = ngpu
        self.main = nn.Sequential(
            # input is (nc) x 64 x 64
            nn.Conv2d(nc, ndf, 4, 2, 1, bias=False),
            nn.LeakyReLU(0.2, inplace=True),
            # state size. (ndf) x 32 x 32
            nn.Conv2d(ndf, ndf * 2, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ndf * 2),
            nn.LeakyReLU(0.2, inplace=True),
            # state size. (ndf*2) x 16 x 16
            nn.Conv2d(ndf * 2, ndf * 4, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ndf * 4),
            nn.LeakyReLU(0.2, inplace=True),
            # state size. (ndf*4) x 8 x 8
            nn.Conv2d(ndf * 4, ndf * 8, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ndf * 8),
            nn.LeakyReLU(0.2, inplace=True),
            # state size. (ndf*8) x 4 x 4
            nn.Conv2d(ndf * 8, 1, 4, 1, 0, bias=False),
            nn.Sigmoid()
        )

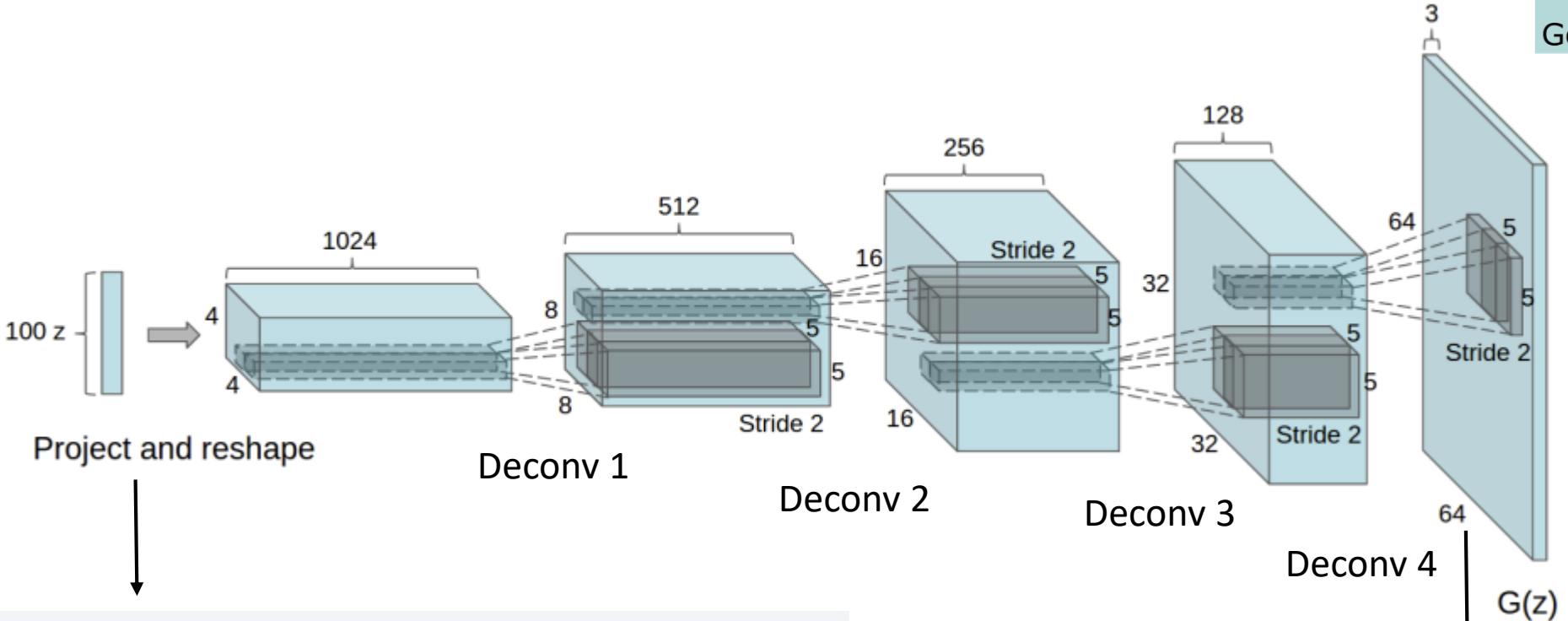
    def forward(self, input):
        return self.main(input)
```

Image credit: https://gluon.mxnet.io/chapter14_generative-adversarial-networks/dcgan.html

Generator

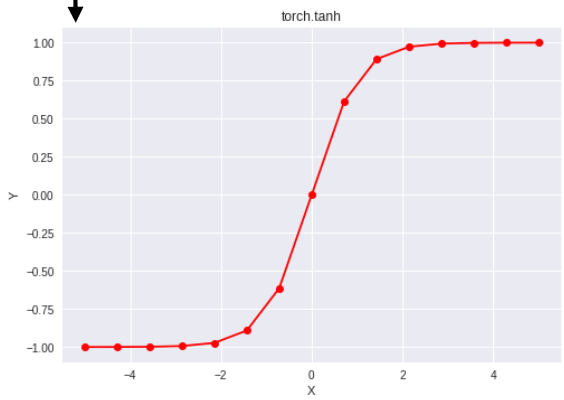
Deep Convolutional GAN (DCGAN)






```
# Generate batch of latent vectors
noise = torch.randn(b_size, nz, 1, 1, device=device)
```

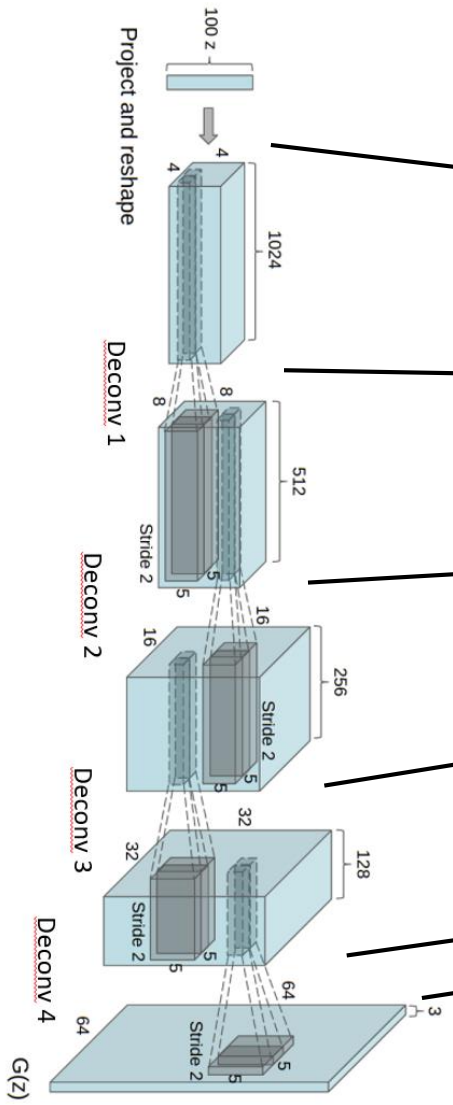
Random numbers from normal distribution with mean 0 and variance 1



Tanh
Normalize to [-1,1]
Consistent with image pixel input range

Image credit: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

Generator – pytorch implementation




```python
class Generator(nn.Module):
    def __init__(self, ngpu):
        super(Generator, self).__init__()
        self.ngpu = ngpu
        self.main = nn.Sequential(
            # input is Z, going into a convolution
            nn.ConvTranspose2d( nz, ngf * 8, 4, 1, 0, bias=False),
            nn.BatchNorm2d(ngf * 8),
            nn.ReLU(True),
            # state size. (ngf*8) x 4 x 4
            nn.ConvTranspose2d(ngf * 8, ngf * 4, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ngf * 4),
            nn.ReLU(True),
            # state size. (ngf*4) x 8 x 8
            nn.ConvTranspose2d( ngf * 4, ngf * 2, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ngf * 2),
            nn.ReLU(True),
            # state size. (ngf*2) x 16 x 16
            nn.ConvTranspose2d( ngf * 2, ngf, 4, 2, 1, bias=False),
            nn.BatchNorm2d(ngf),
            nn.ReLU(True),
            # state size. (ngf) x 32 x 32
            nn.ConvTranspose2d( ngf, nc, 4, 2, 1, bias=False),
            nn.Tanh()
            # state size. (nc) x 64 x 64
        )

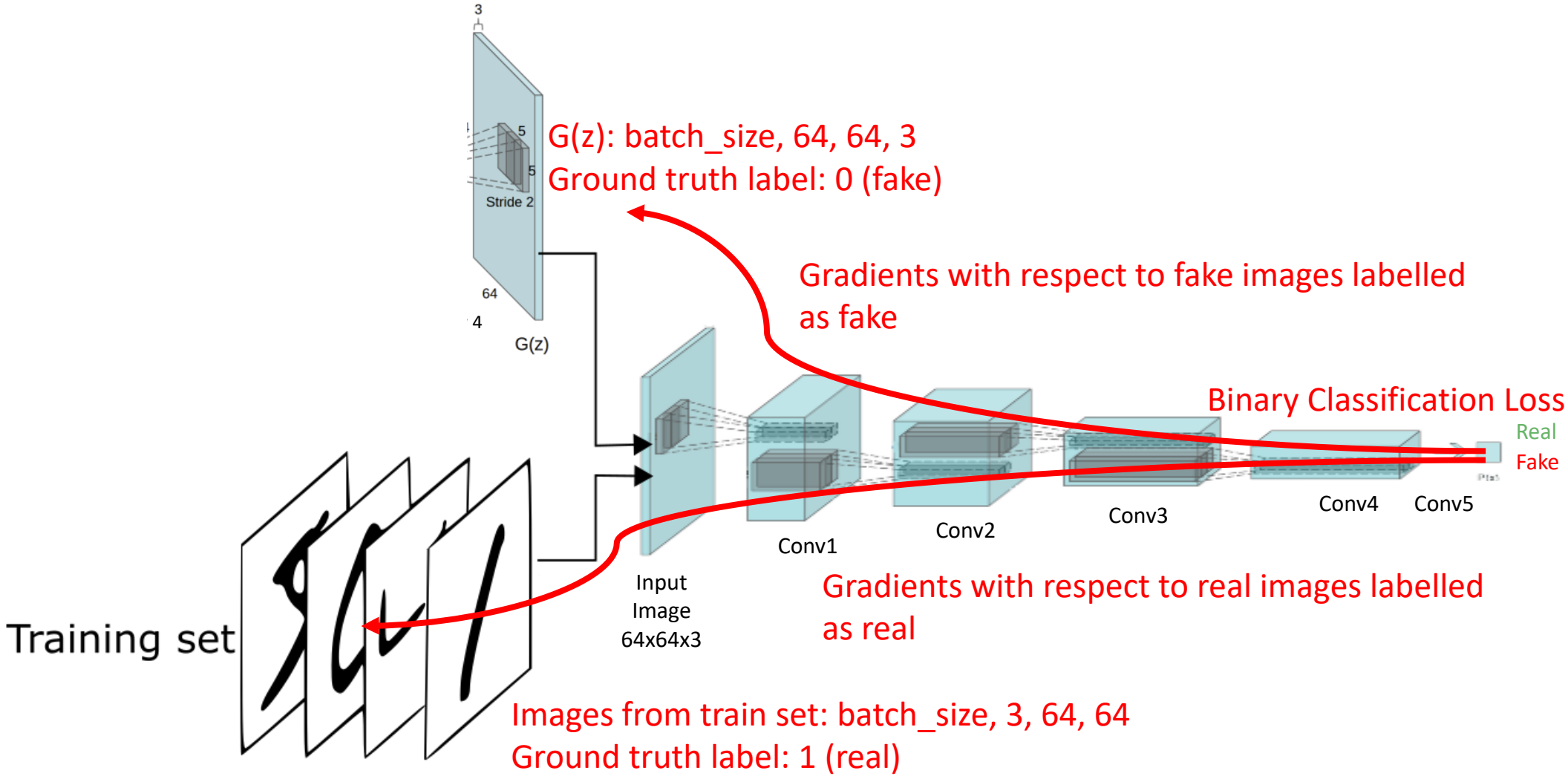
    def forward(self, input):
        return self.main(input)
```

Training GAN – Part 1

Generator  Discriminator

Min-max Game

Training part 1: train discriminator

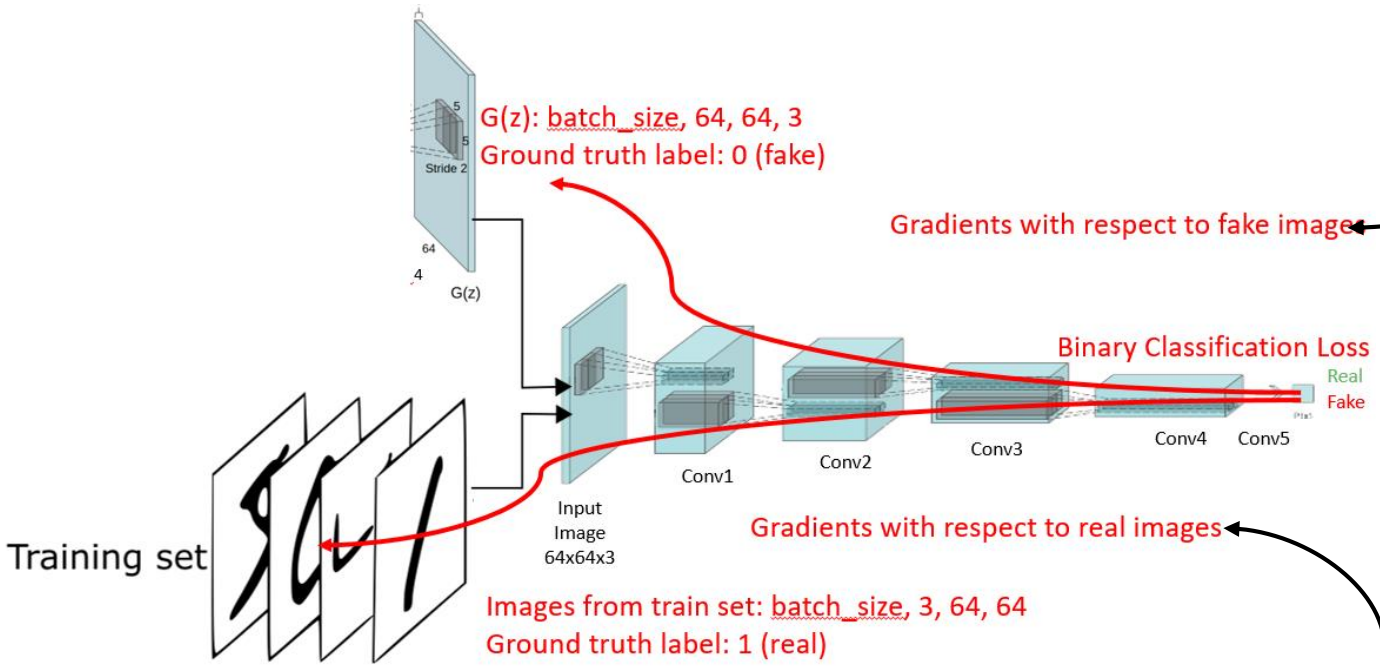


Training GAN – Part 1

Generator  Discriminator

Min-max Game

Training part 1: train discriminator




```
## Train with all-fake batch
# Generate batch of latent vectors
noise = torch.randn(b_size, nz, 1, 1, device=device)
# Generate fake image batch with G
fake = netG(noise)
label.fill_(fake_label)
# Classify all fake batch with D
output = netD(fake.detach()).view(-1)
# Calculate D's loss on the all-fake batch
errD_fake = criterion(output, label)
# Calculate the gradients for this batch
errD_fake.backward()
D_G_z1 = output.mean().item()
# Add the gradients from the all-real and all-fake batches
errD = errD_real + errD_fake
# Update D
optimizerD.step()
```

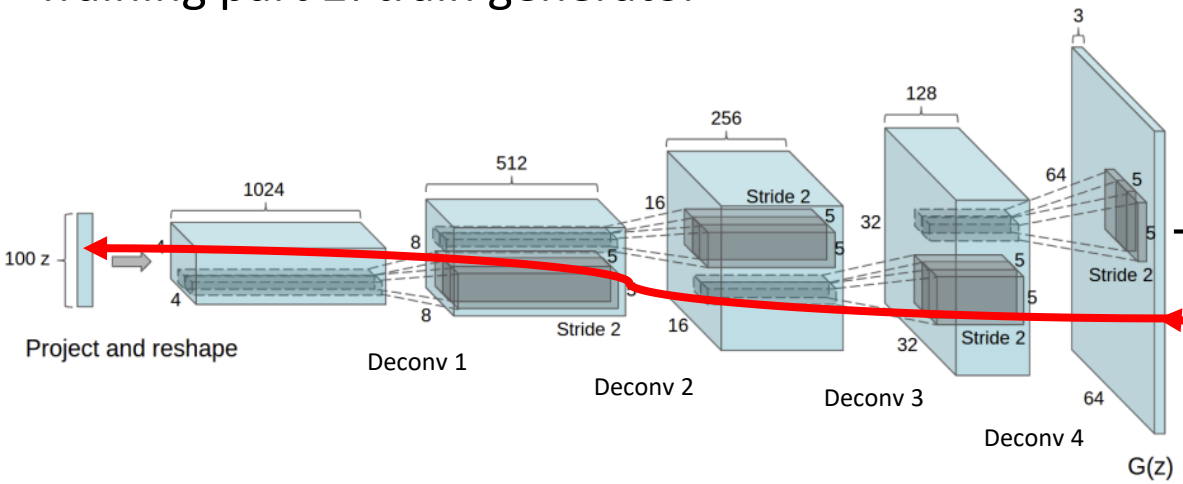
```
## Train with all-real batch
netD.zero_grad()
# Format batch
real_cpu = data[0].to(device)
b_size = real_cpu.size(0)
label = torch.full((b_size,), real_label, device=device)
# Forward pass real batch through D
output = netD(real_cpu).view(-1)
# Calculate loss on all-real batch
errD_real = criterion(output, label)
# Calculate gradients for D in backward pass
errD_real.backward()
D_x = output.mean().item()
```

Training GAN – Part 2

Generator  Discriminator

Min-max Game

Training part 2: train generator

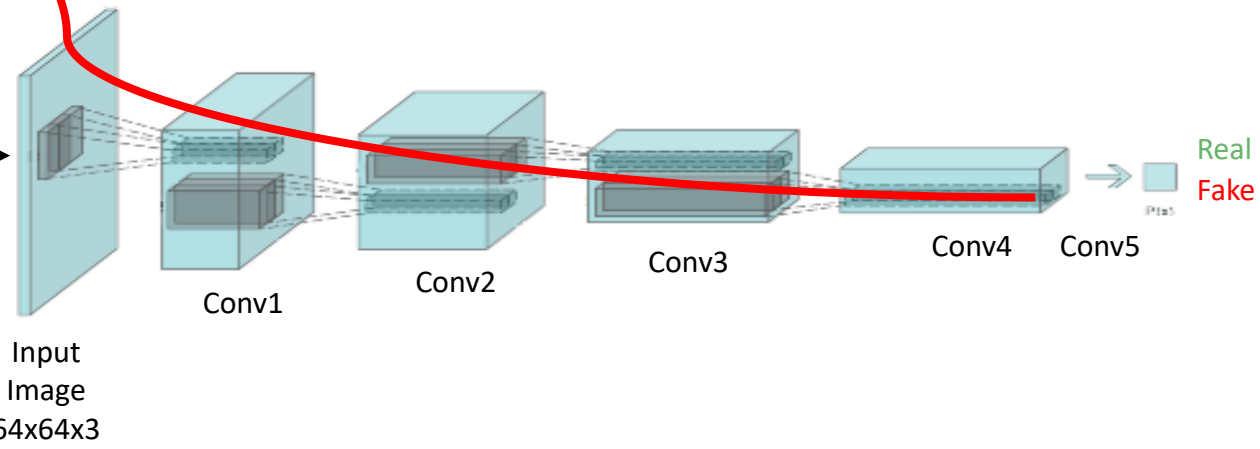


G(z): batch_size, 64, 64, 3

Ground truth label: 1 (real)

**In discriminator training, the ground truth label: 0 (fake)

Gradients with respect to fake images but labelled as “real”

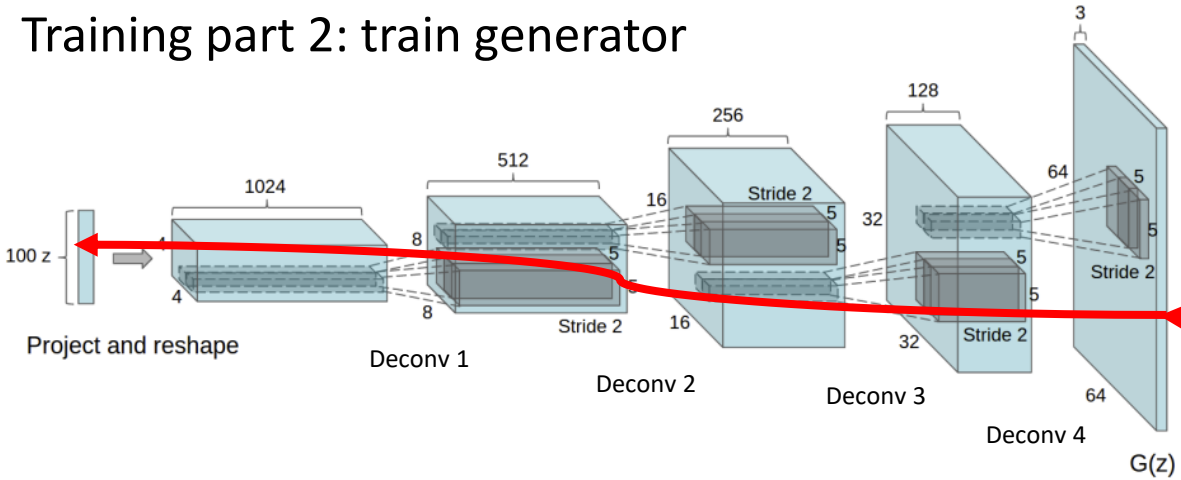


Training GAN – Part 2

Generator  Discriminator

Min-max Game

Training part 2: train generator

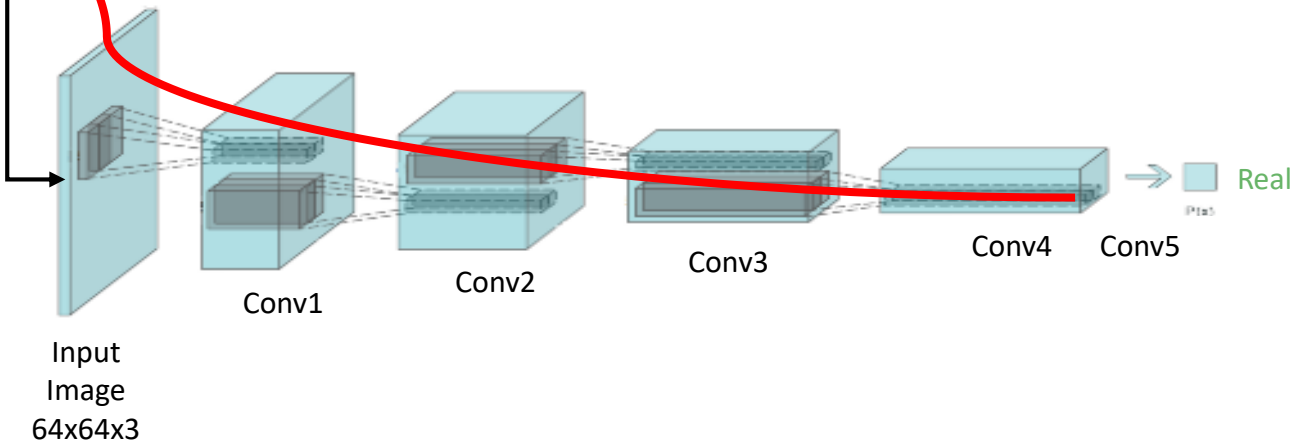


G(z): batch_size, 64, 64, 3

Ground truth label: 1 (real)

**In discriminator training, the ground truth label: 0 (fake)

Gradients with respect to fake images but labelled as “real”




```
##############################
# (2) Update G network: maximize log(D(G(z)))
##############################
netG.zero_grad()
label.fill_(real_label)  # fake labels are real for generator cost
# Since we just updated D, perform another forward pass of all-fake batch through D
output = netD(fake).view(-1)
# Calculate G's loss based on this output
errG = criterion(output, label)
# Calculate gradients for G
errG.backward()
D_G_z2 = output.mean().item()
# Update G
optimizerG.step()
```

GAN Zoo

cGAN

StyleGAN

CycleGAN

ProgressiveGAN

InforGAN

LapGAN

BigBiGAN

AC-GAN

BiGAN

StackGAN

EBGAN

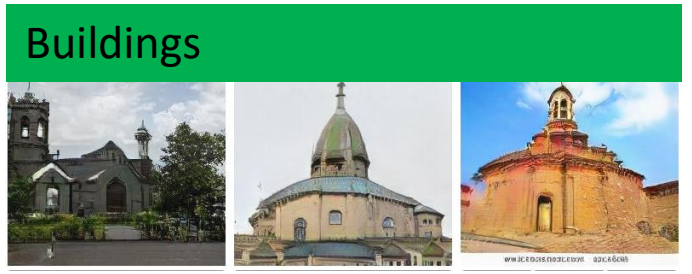
WGAN

BigGAN

BEGAN

Problems with Deep Convolutional GAN (DCGAN)

- Generated images are very small, 64x64, 128x128
- One generator only corresponds with one class of images (no control over random vector z)
- Generated image quality is bad
- Training GAN is brittle:
  - Non-convergence: Model parameter oscillate and never converge
  - Model collapse: Produce limited number of samples
  - Diminished gradients: discriminator is too perfect and generator always fails
  - Highly sensitive to hyperparameters

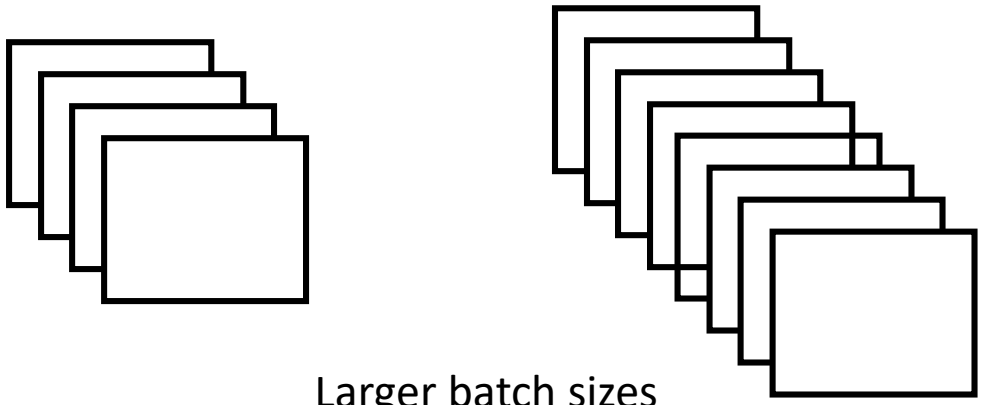




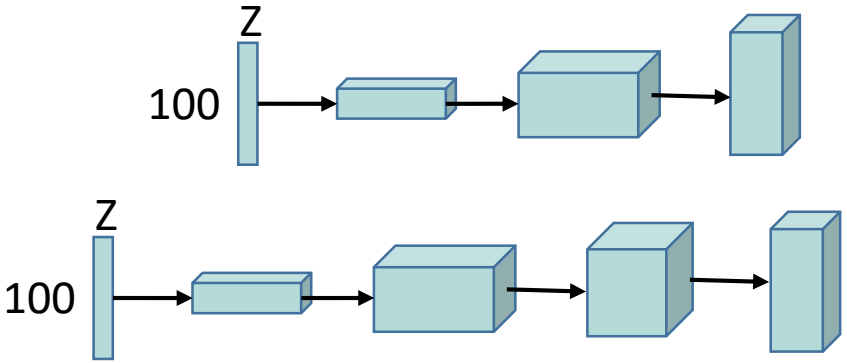
https://machinelearningmastery.com/a-gentle-introduction-to-the-biggan/

BigBiGAN

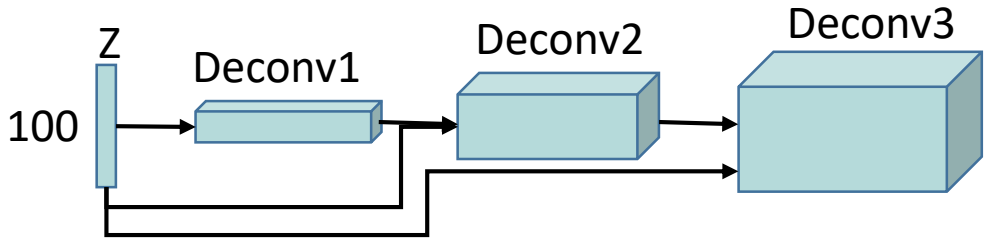
Larger batch size, more network parameters, network architecture designs (skip z-connections, self-attention)



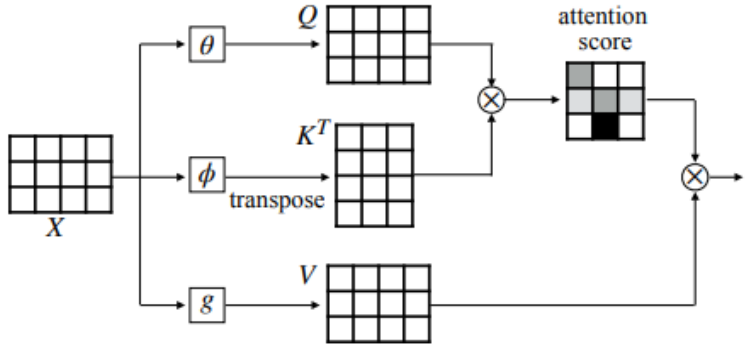
Larger batch sizes



More network parameters



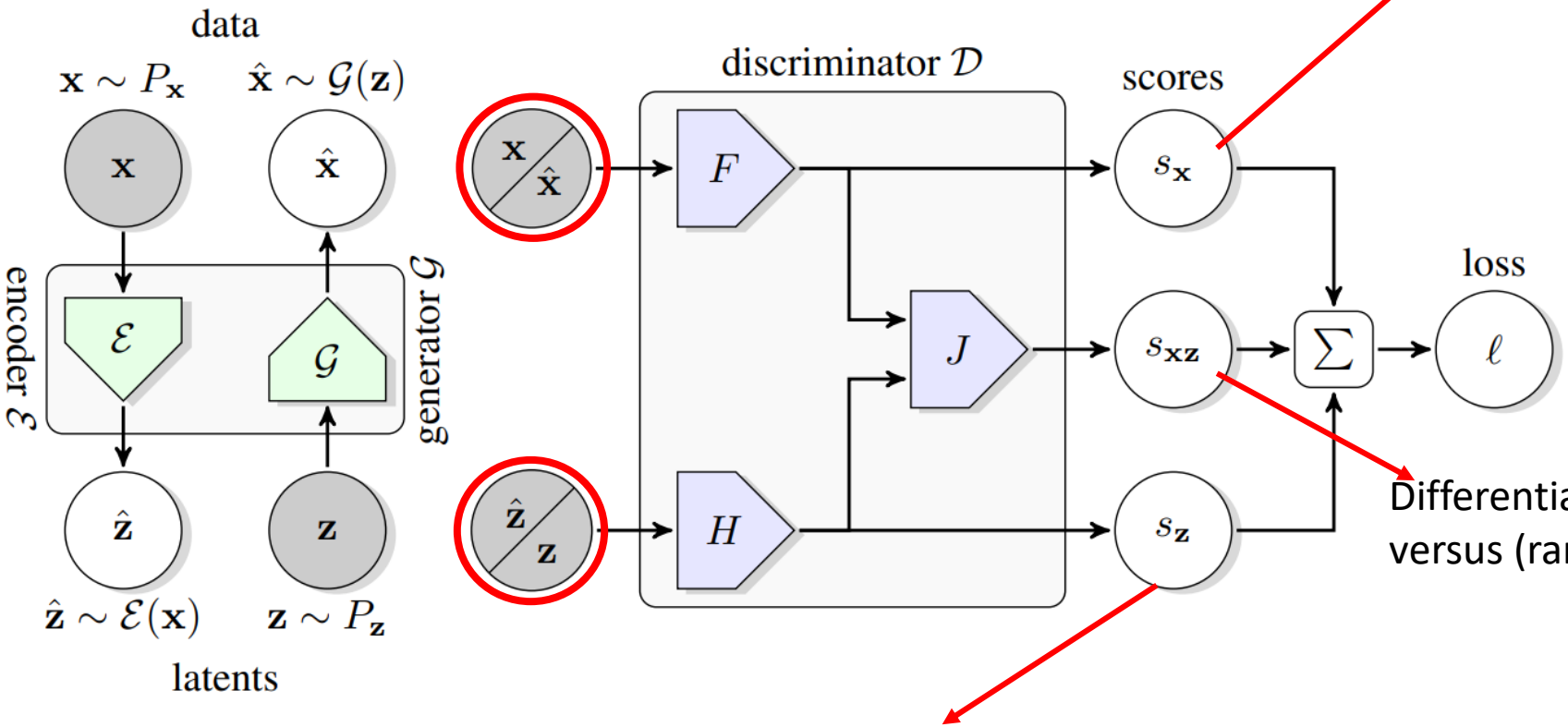
Skip z-connections in generator



Self-attention Module in GAN

Donahue etal, Large Scale Adversarial Representation Learning, NIPS, 2019

BigBiGAN



Differentiate generated images from real images;

Differentiate pair (encoded z, real images) versus (randomly sampled z, generated images)

Differentiate encoded z from real images and randomly sampled z

X: input image
Z: latent code

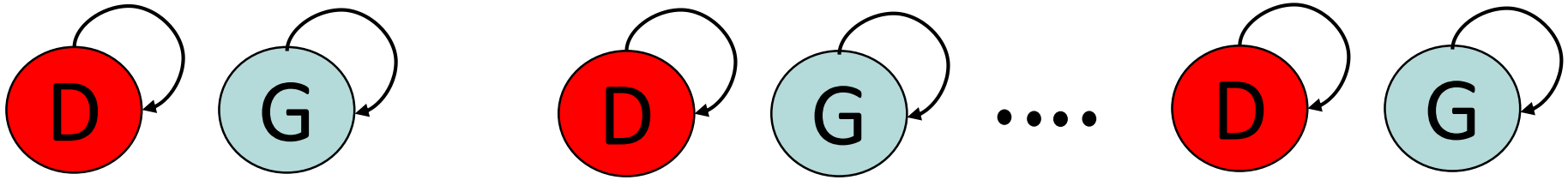
Donahue etal, Large Scale Adversarial Representation Learning, NIPS, 2019

Tricks for More Realistic Image Reconstruction

- Update discriminators more often than generators during training



- Moving average of model weights (Progressive GAN)



After 1st epoch, Generator Parameter W_G1

After 2nd epoch, Generator Parameter W_G2

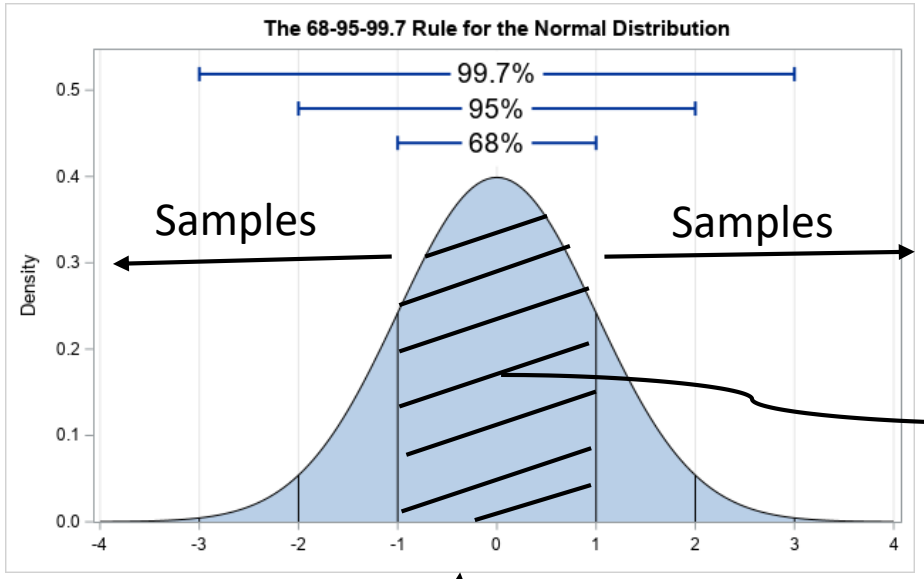
After Tth epoch, Generator Parameter W_GT

W_Gfinal = average(W_G1, W_G2, … ,W_GT)

https://machinelearningmastery.com/a-gentle-introduction-to-the-biggan/

Tricks for More Realistic Image Reconstruction

- Truncate z resampling at test stage



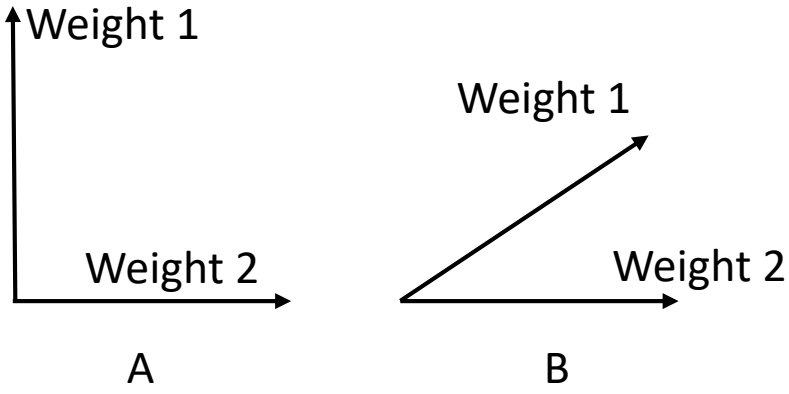
Sacrifice
sample diversity

Z

100

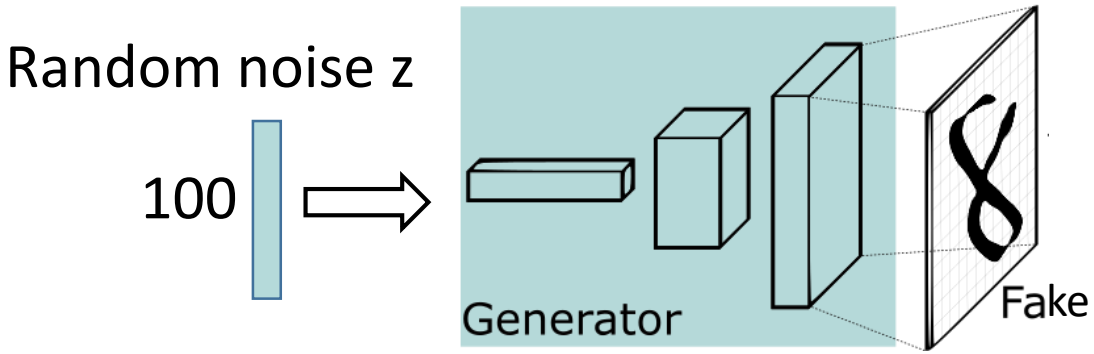
```
# Generate batch of latent vectors
noise = torch.randn(b_size, nz, 1, 1, device=device)
```

- Orthogonal weight initialization
- Orthogonal weight regularization



https://machinelearningmastery.com/a-gentle-introduction-to-the-biggan/

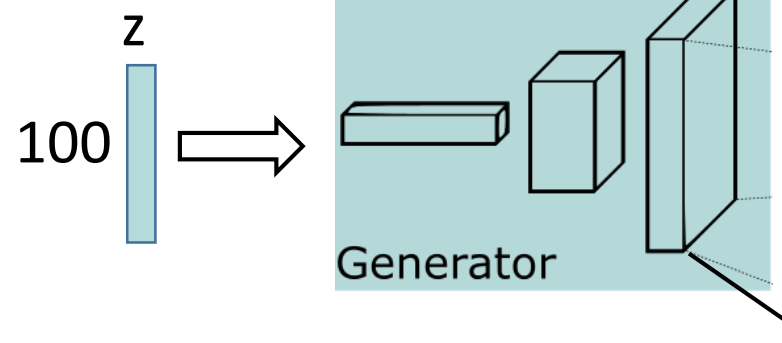
Brain Reading



Brain Codes from
ECoG, fMRI, EEG, MEG, etc



All Animal species



Five senses: Smell, Images
Touch, Sound, Taste

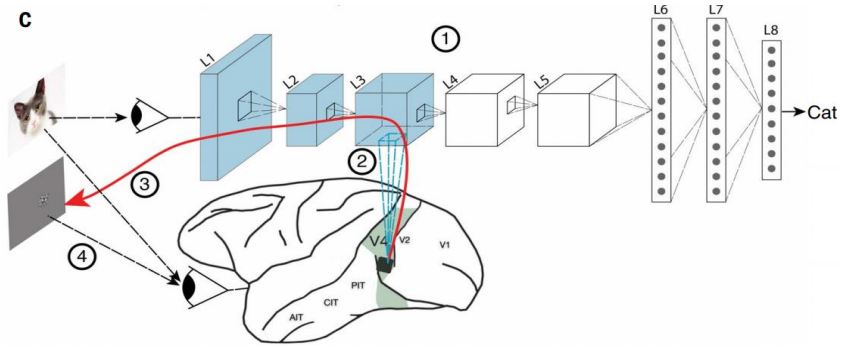
High-level functionality:
Emotions, Memories,
Language

Image Reconstruction Methods from Brain Signals

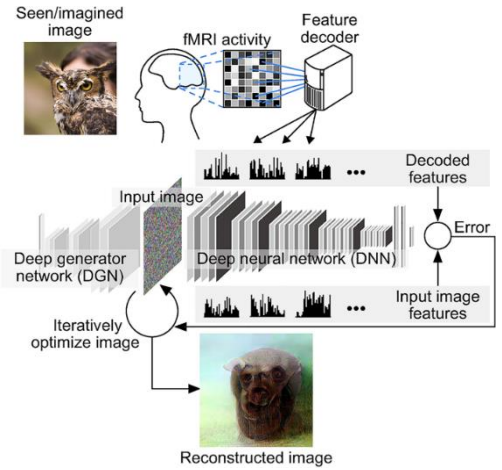
- Overview of Image Reconstruction Methods from Brain Signals
  - Gradient Backpropogation
    - DeepDream
    - TextureSynthesis, StyleTransfer

They are NOT generative models; but still cool

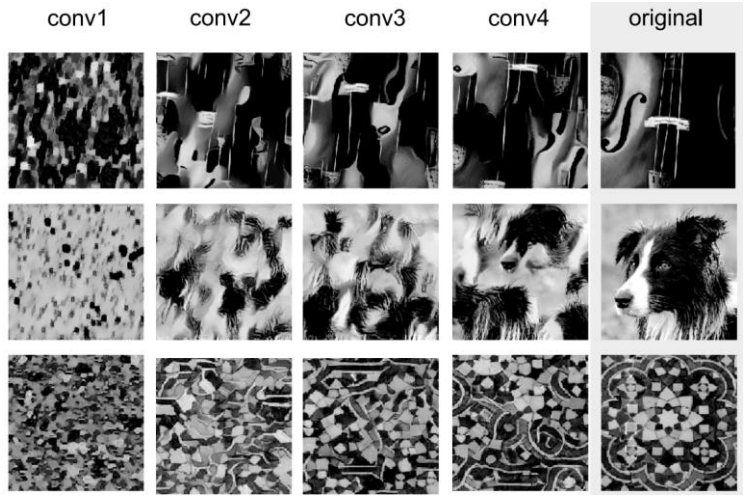
Gradient Backpropogation on pre-trained object recognition networks



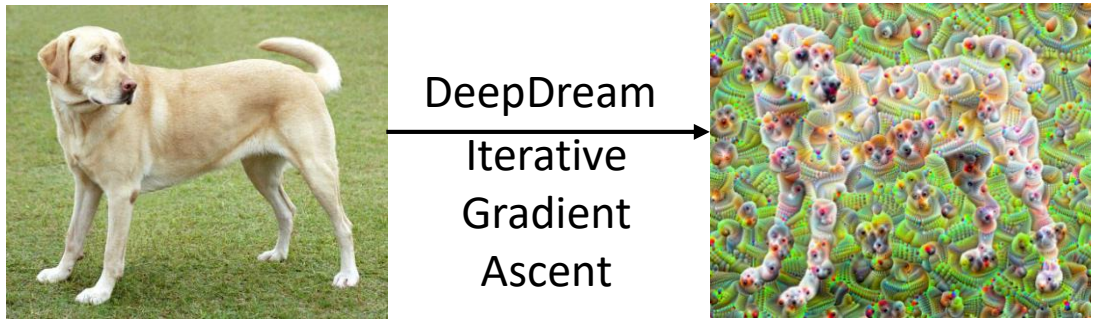
Bashivan etal, Science, 2019



Shen etal, Plos Computational Biology, 2019



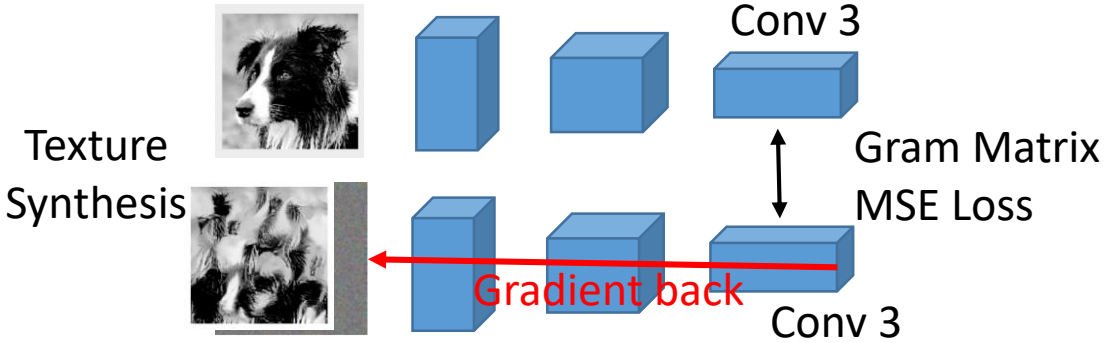
Cadena etal, Plos Computational Biology, 2019



DeepDream

Iterative Gradient Ascent

Feature MSE Loss at target layer



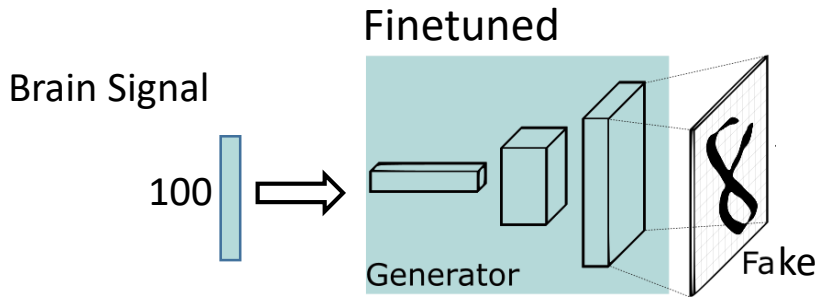
Code link: https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/generative/deepdream.ipynb#scrollTo=XPDKhwPcNbG7

Gatys etal, NIPS, 2015

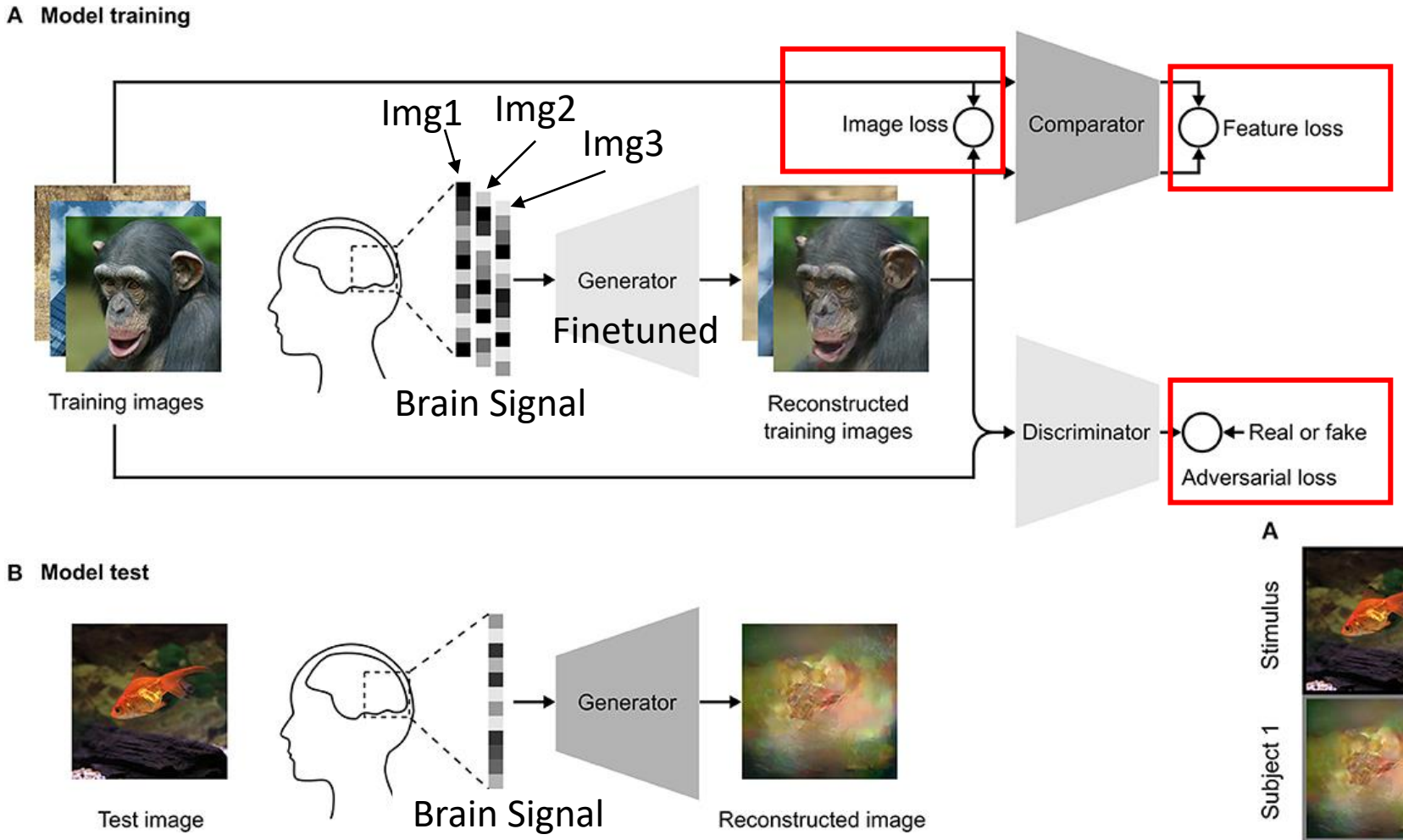
Image Reconstruction Methods from Brain Signals

Overview of Image Reconstruction Methods from Brain Signals

- Gradient Backpropogation (DeepDream, TextureSynthesis, StyleTransfer)
- Generative model (Latent code)
  - Learnable Generative Model (Model parameter is fine-tuned)



End-to-End Learnable Generative Model



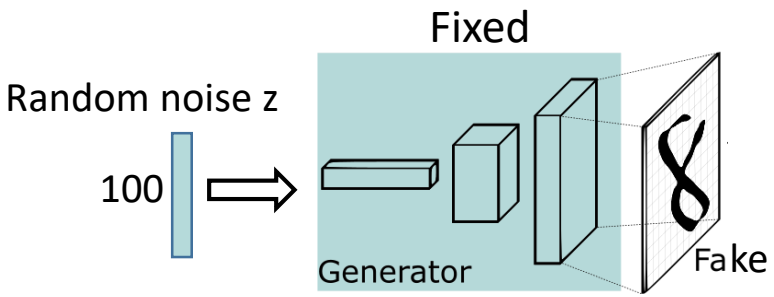


Shen etal, Frontiers in Computational Neuroscience, 2019

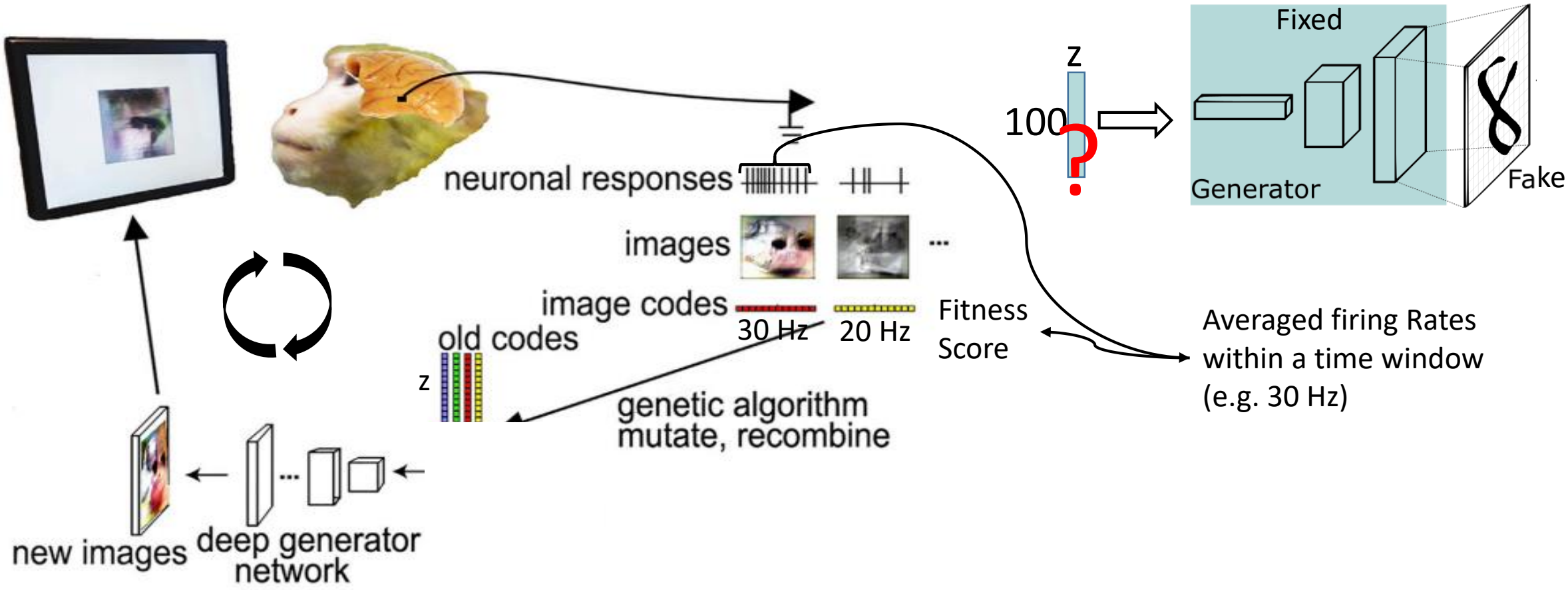
Image Reconstruction Methods from Brain Signals

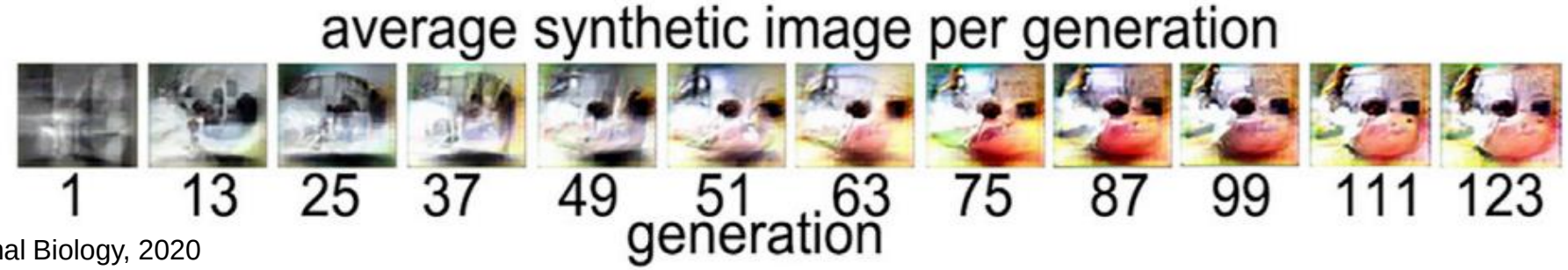
Overview of Image Reconstruction Methods from Brain Signals

- Gradient Backpropogation (DeepDream, TextureSynthesis, StyleTransfer)
- Generative model (Latent code)
  - Learnable Generative Model (Model parameter is fine-tuned)
  - Fixed Generative Model (Model parameter is fixed)
    - Genetic Algorithm



Evolving Latent Code using Genetic Algorithm

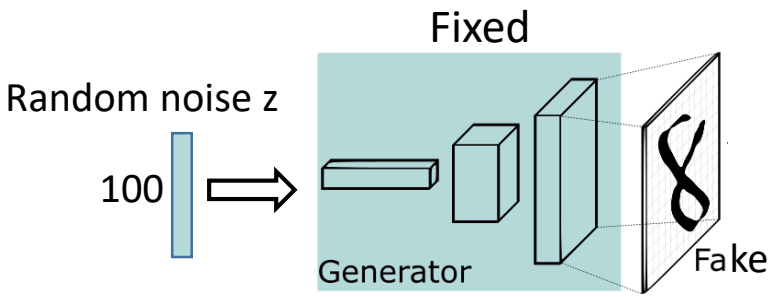




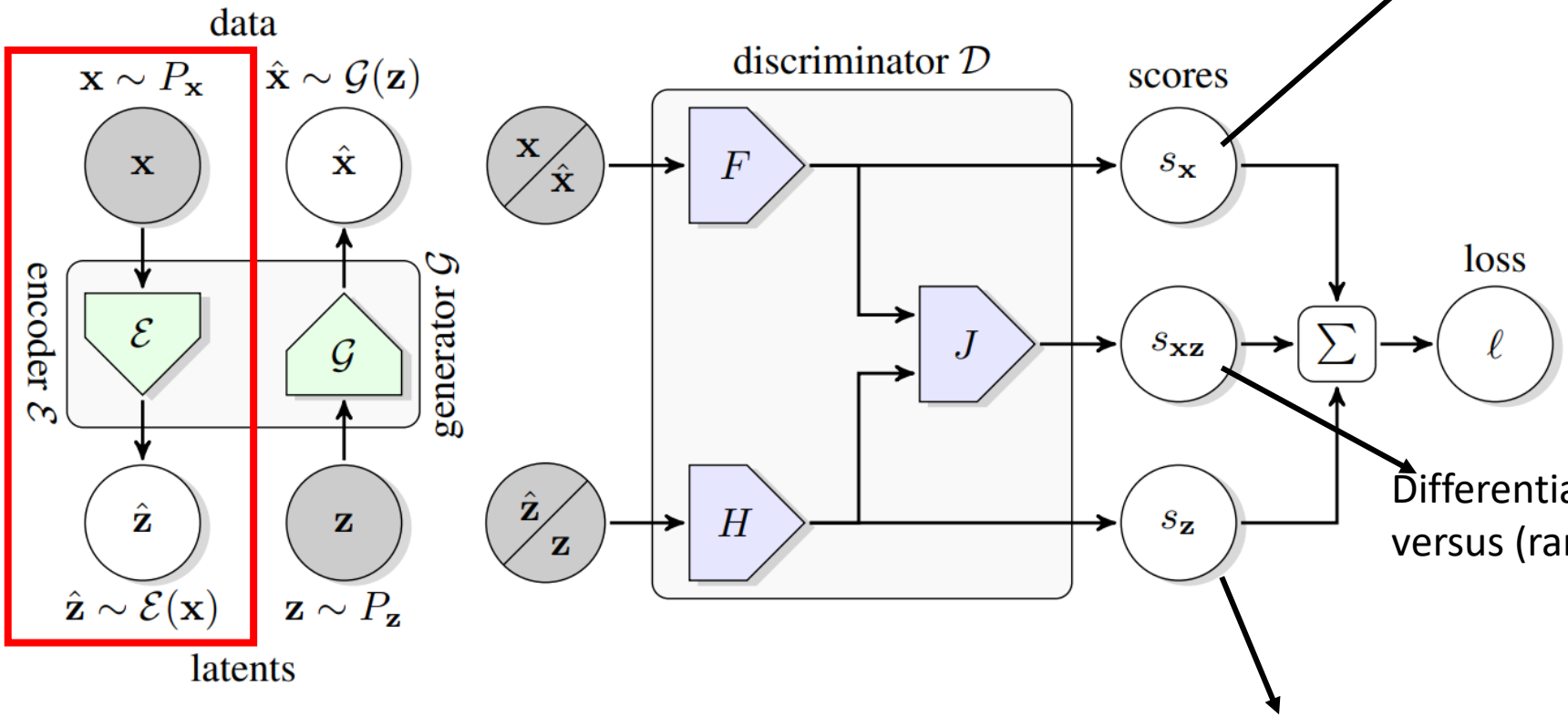
Ponce etal, Cell, 2019; Xiao etal, Plos Computational Biology, 2020

Image Reconstruction Methods from Brain Signals

- Overview of Image Reconstruction Methods from Brain Signals
  - Gradient Backpropogation (DeepDream, TextureSynthesis, StyleTransfer)
  - Generative model (Latent code)
    - Learnable Generative Model (Model parameter is fine-tuned)
    - Fixed Generative Model (Model parameter is fixed)
      - Genetic Algorithm
      - **Linear Regression**



BigBiGAN



Differentiate generated images from real images;

Differentiate pair (encoded z, real images) versus (randomly sampled z, generated images)

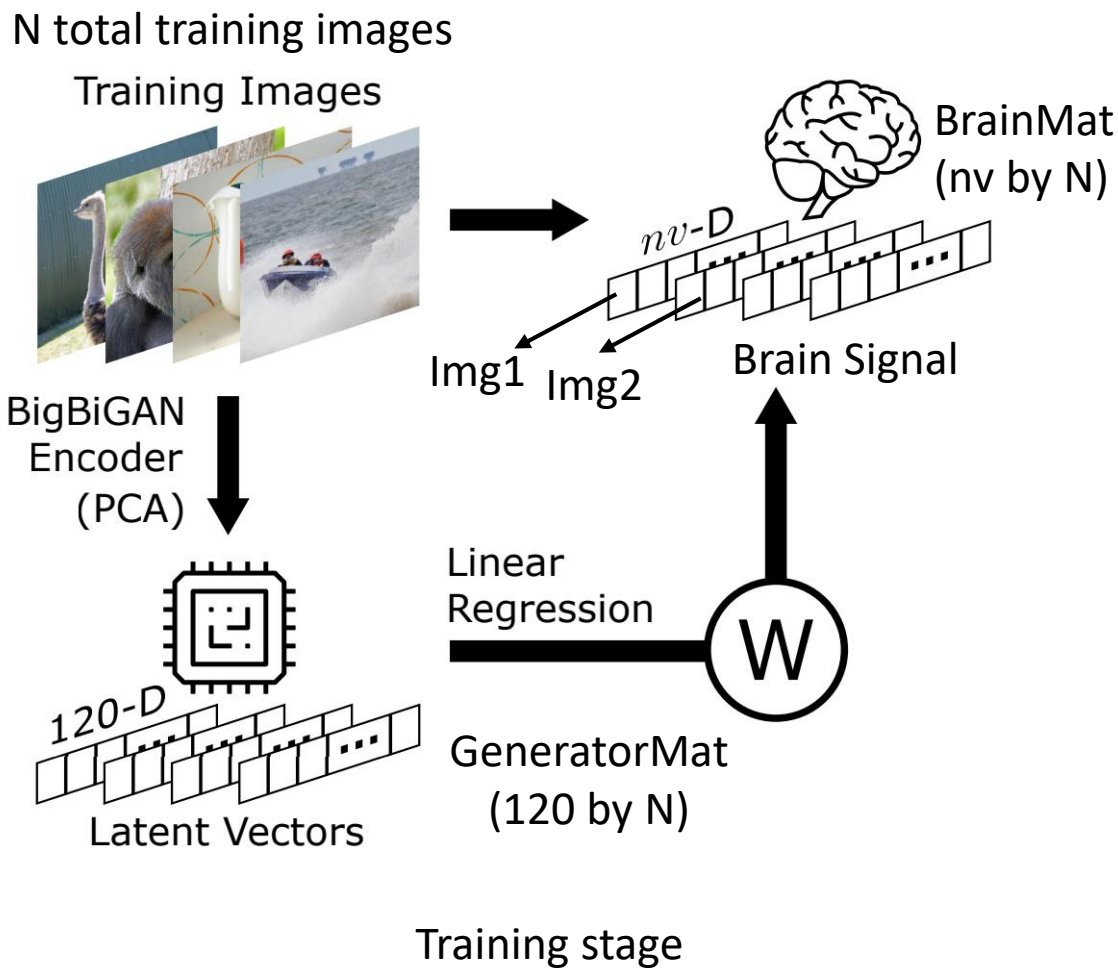
Differentiate encoded z from real images and randomly sampled z

X: input image

Z: latent code

Donahue etal, Large Scale Adversarial Representation Learning, NIPS, 2019

Mapping Latent Code using Linear Regression



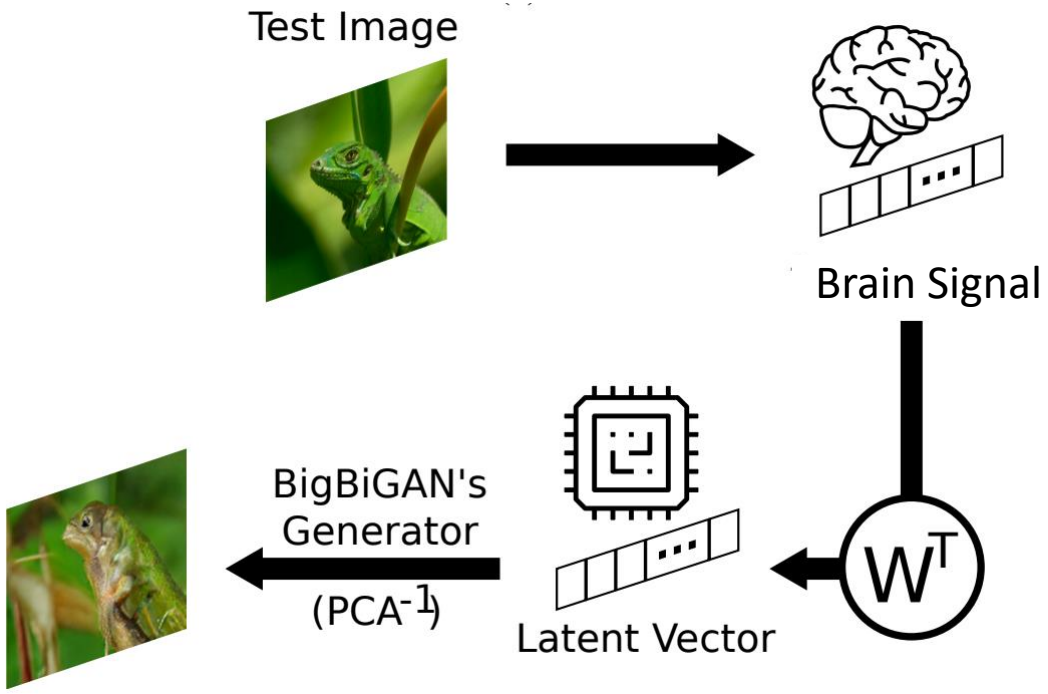
W	GeneratorMat	=	BrainMat
(nv by 120)	(120 by N)		(nv by N)

Practice:

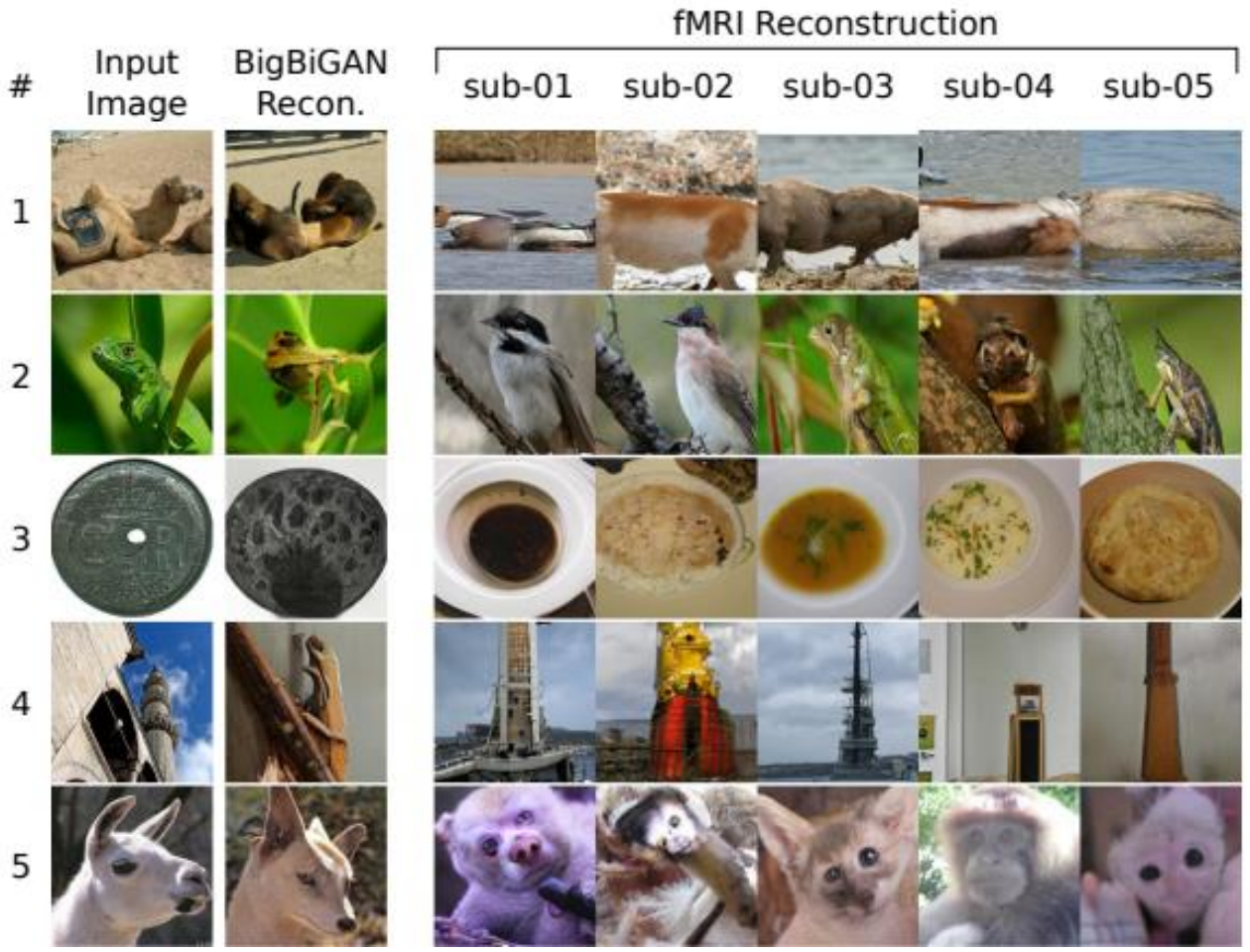
1. W might be invertable: psuedo-inverse
2. BrainMat size is too large: dimension reduction using PCA to pre-process the data
3. Latent code normalization

Mozafari etal, Reconstructing Natural Scenes from fMRI Patterns using BigBiGAN, 2020

Mapping using Linear Regression

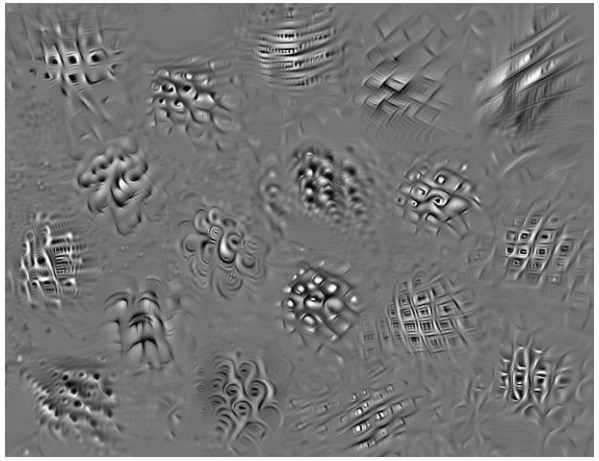


Testing stage

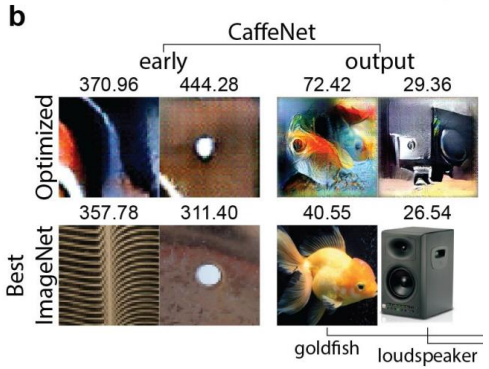


Mozafari etal, Reconstructing Natural Scenes from fMRI Patterns using BigBiGAN, 2020

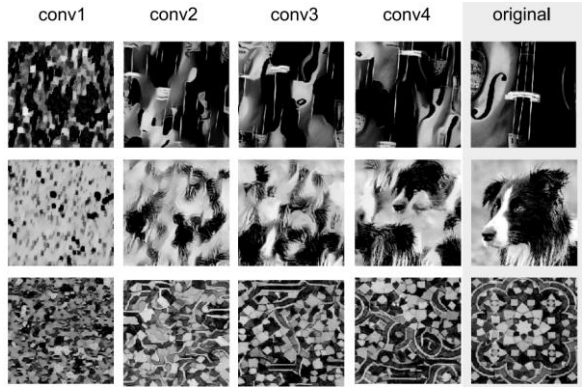
Which brain reader is better?



Bashivan etal, Science, 2019



Ponce etal, Cell, 2019;
Xiao etal, Plos Computational Biology, 2020



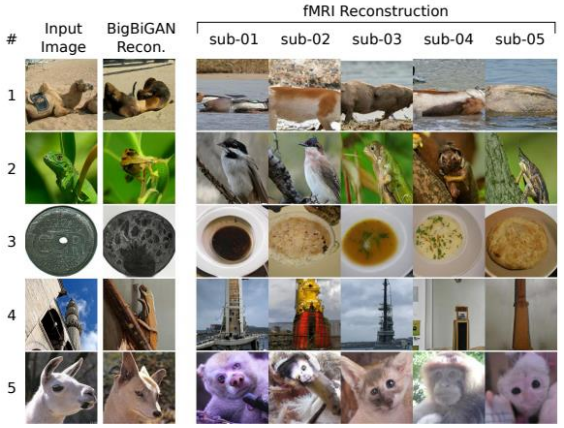
Cadena etal, Plos Computational Biology, 2019



Shen etal, Frontiers in Computational Neuroscience, 2019



Shen etal, Plos Computational Biology, 2019



Mozafari etal, arxiv, 2020

Do not be subjective. Use quantative metrics to evaluate image reconstruction quality

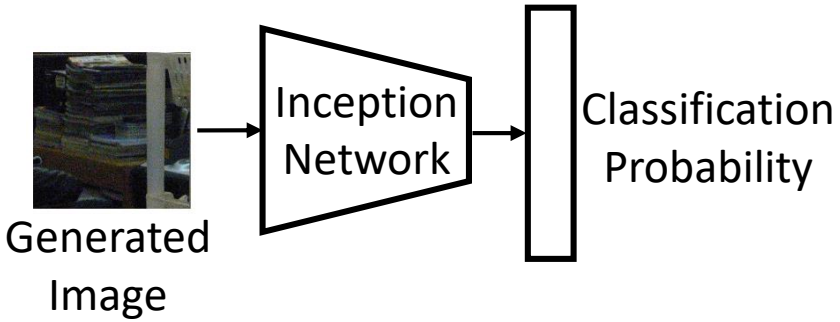
More brain scores on generative models



Real Reconstructed

Methods	L2 norm, Cosine SIM, etc At pixel level	Inception Score (IS)	Frechet Inception Distance (FID)	ImageNet Classification Accuracy	Human Behavioral Score	Brain Signal Score (Linear Correlation)
Bashivan etal, Science, 2019						
Shen etal, Frontiers in Computational Neuroscience, 2019						
Mozafari etal, arxiv, 2020						
Ponce etal, Cell, 2019; Xiao etal, Plos Computational Biology, 2020						
Shen etal, Plos Computational Biology, 2019						
Cadena etal, Plos Computational Biology, 2019						
Many more to come …						

More brain scores - IS



* Inception Score (measuring image quality + diversity)

Image Quality (more focused label distribution -> better; lower entropy)

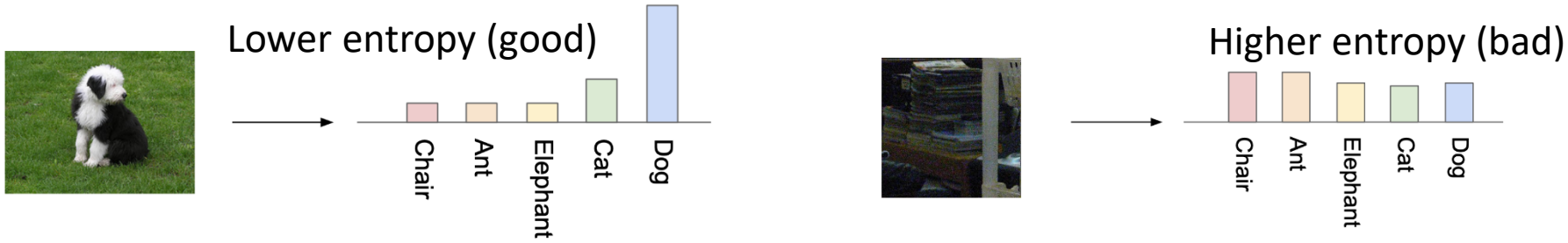


Image Diversity (more uniform marginal distribution -> better; higher entropy)

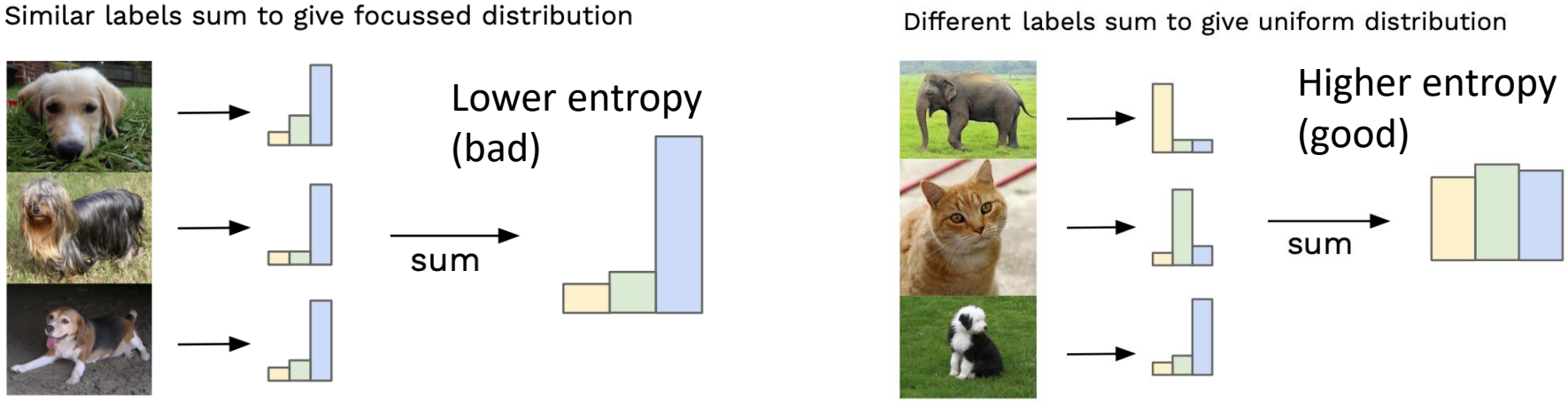
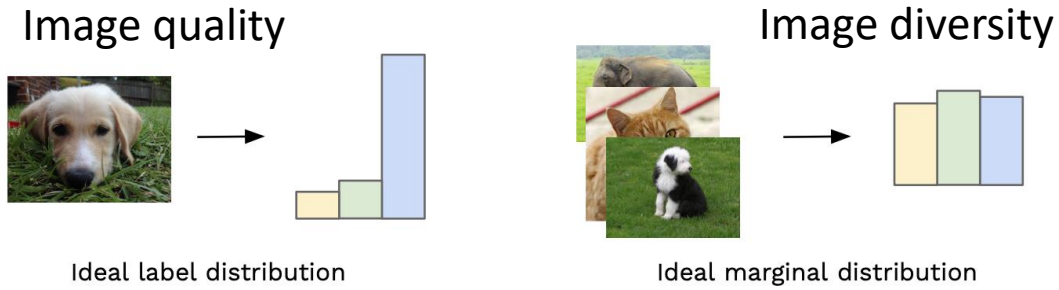
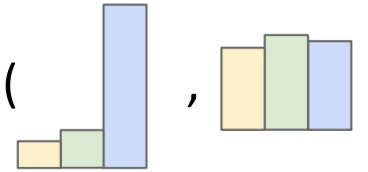


Image credit: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

More brain scores - FID

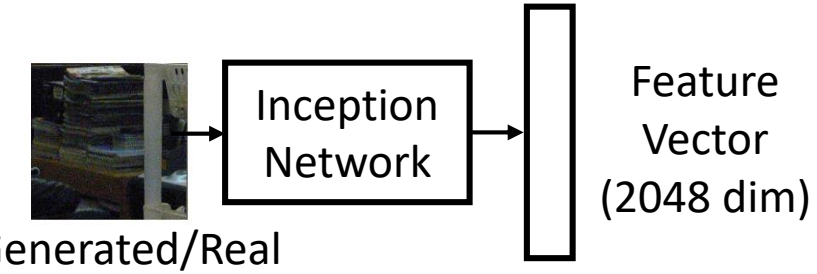
* Inception Score (measuring image quality + diversity)

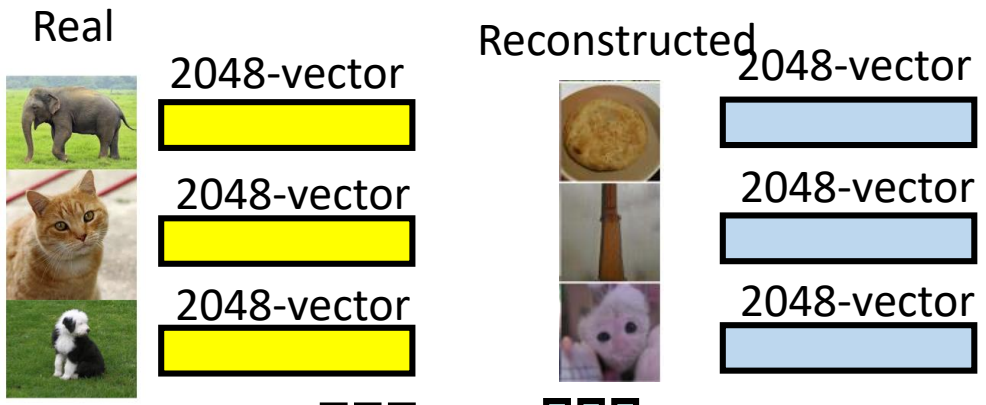


Inception Score (IS) = KL Divergence ()

The higher IS score, the better.

* Fréchet Inception Distance (FID)





FID= Distance between two multi-variant Gaussian distribution ( , )

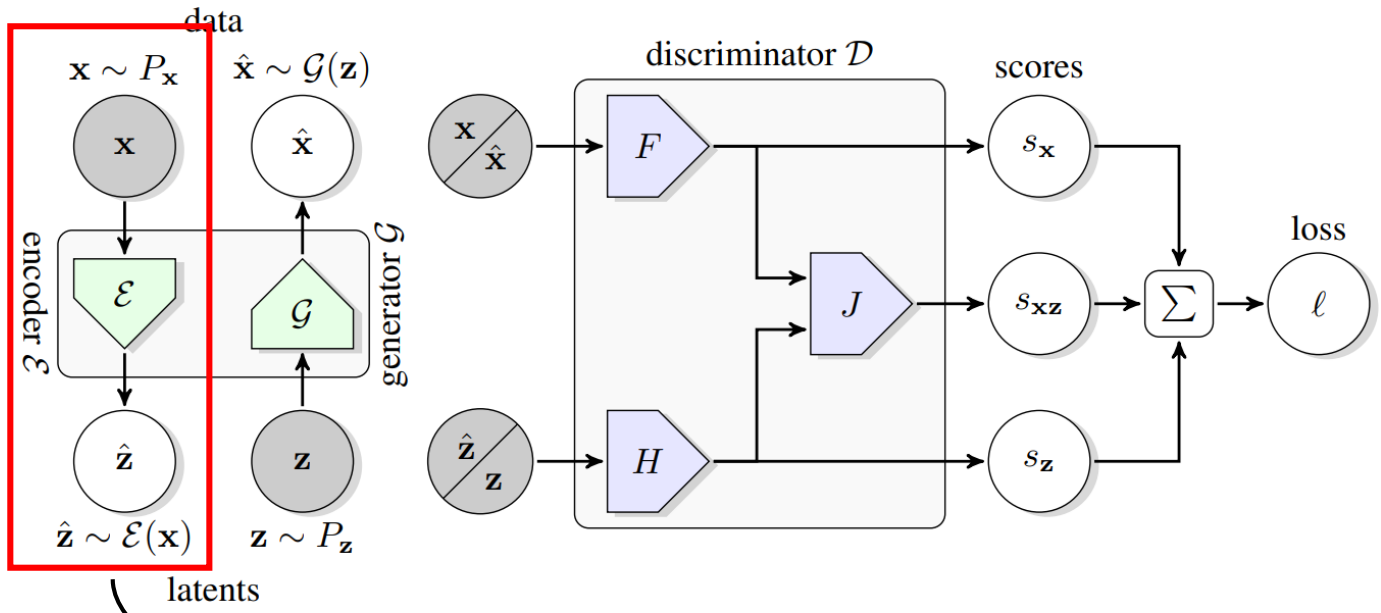
The lower FID score, the better.

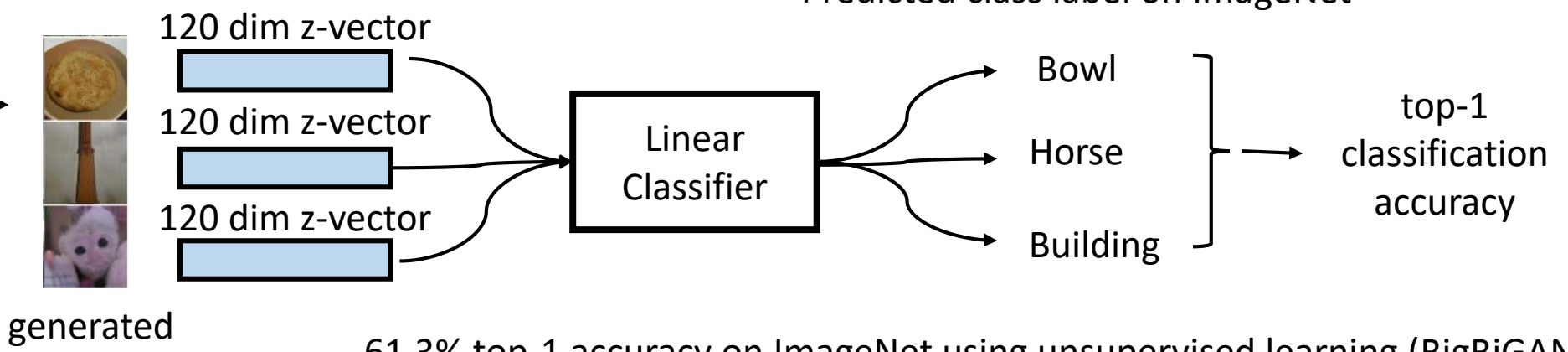
$$\mathrm{FID} = ||\mu_r - \mu_g||^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}),$$

Image credit: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

More brain scores – Classification accuracy

* ImageNet top-1 classification accuracy

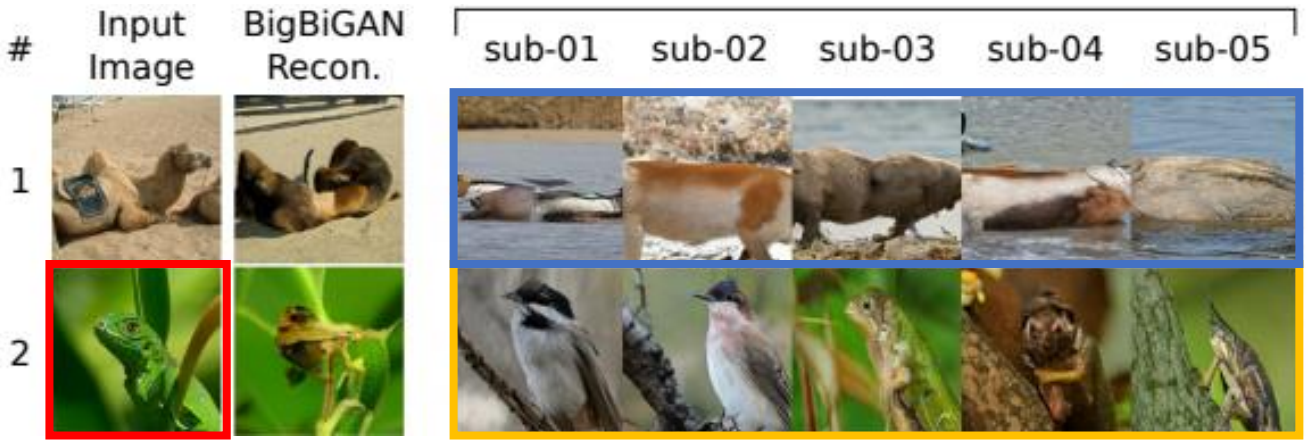




61.3% top-1 accuracy on ImageNet using unsupervised learning (BigBiGAN)!

More brain scores

*Human Judgment at behavioral levels





Real Image as target

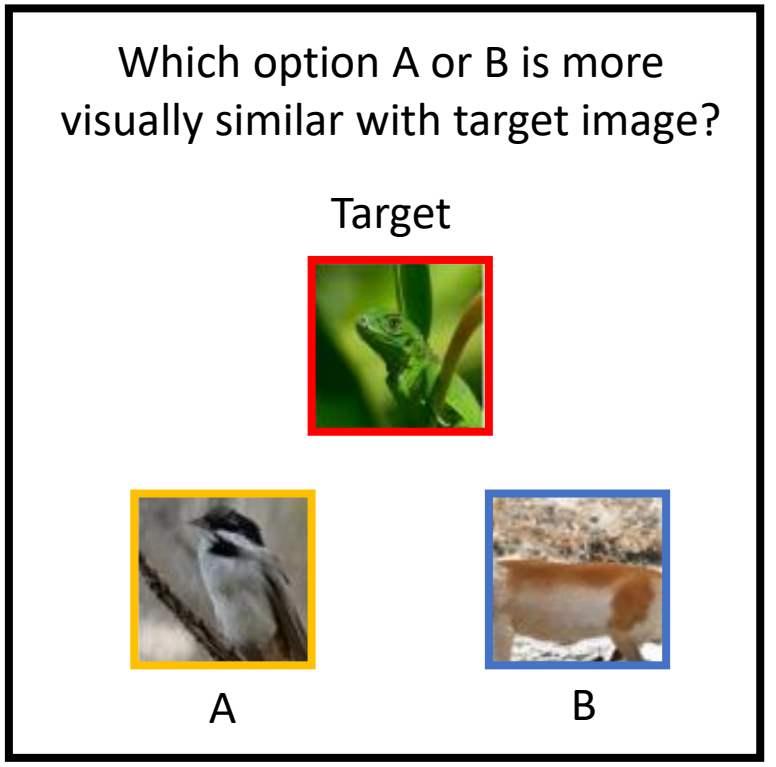


Randomly chosen reconstructed image based on brain activity from real image

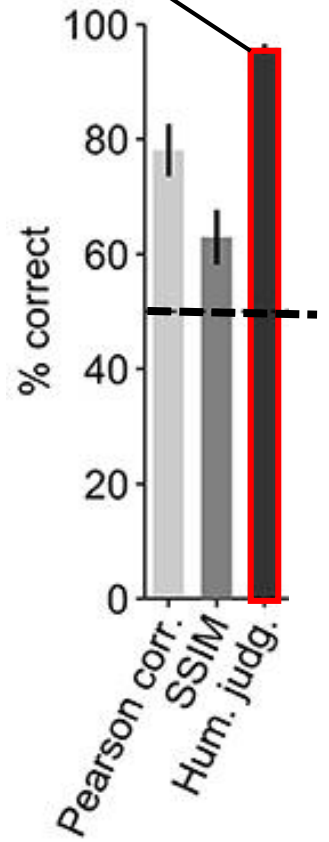


Randomly chosen reconstructed image from other irrelevant images

~90% people chose options which are reconstructed based on brain signals generated after seeing target image



Example trial on Amazon Mechanical Turk

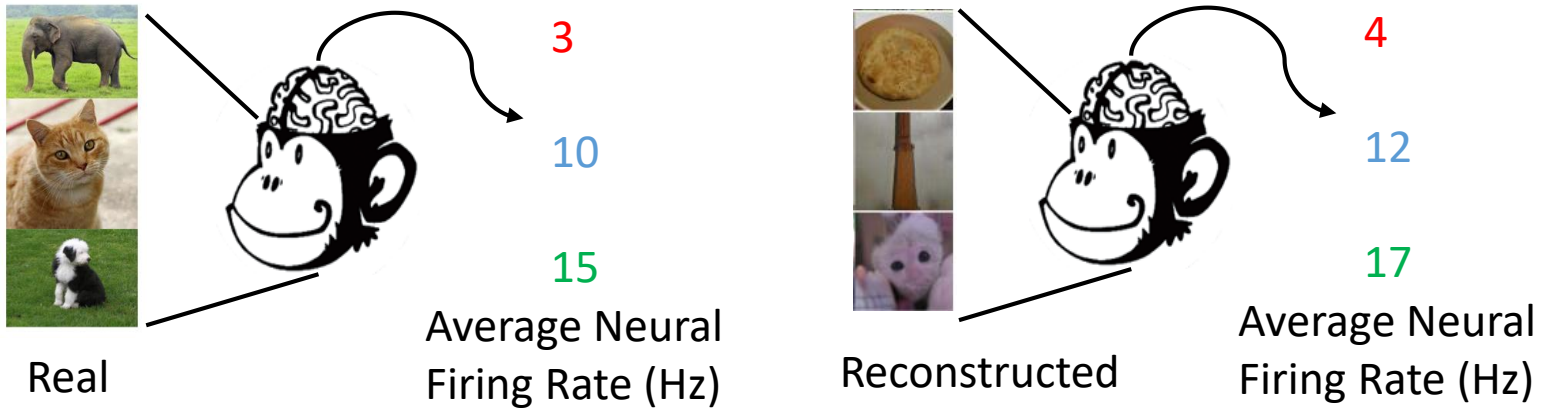


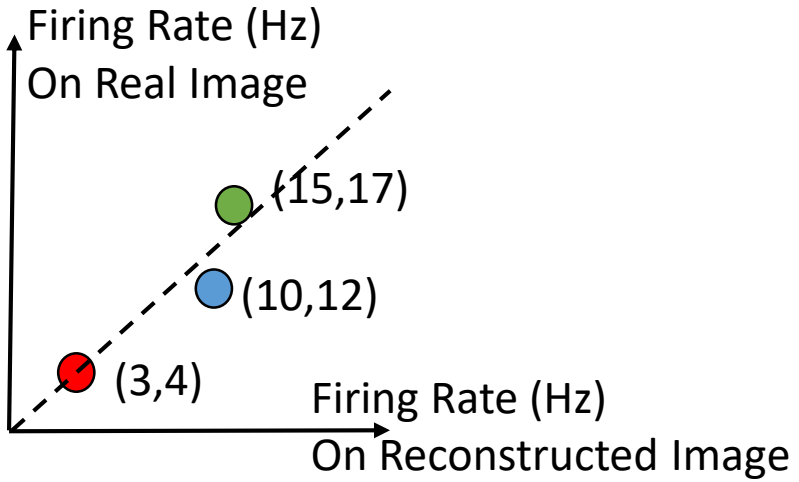
Shen etal, Frontiers in Computational Neuroscience, 2019

More brain scores

*Control ability of brain signals:

<u>Linear correlation</u> between predicted brain signals and real brain signal

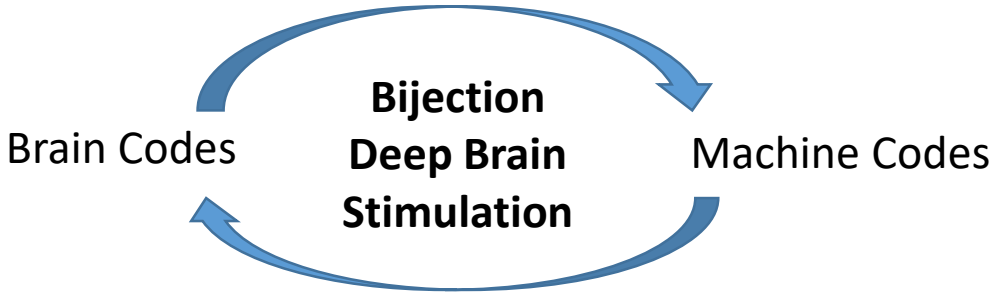




Any other brain signals …

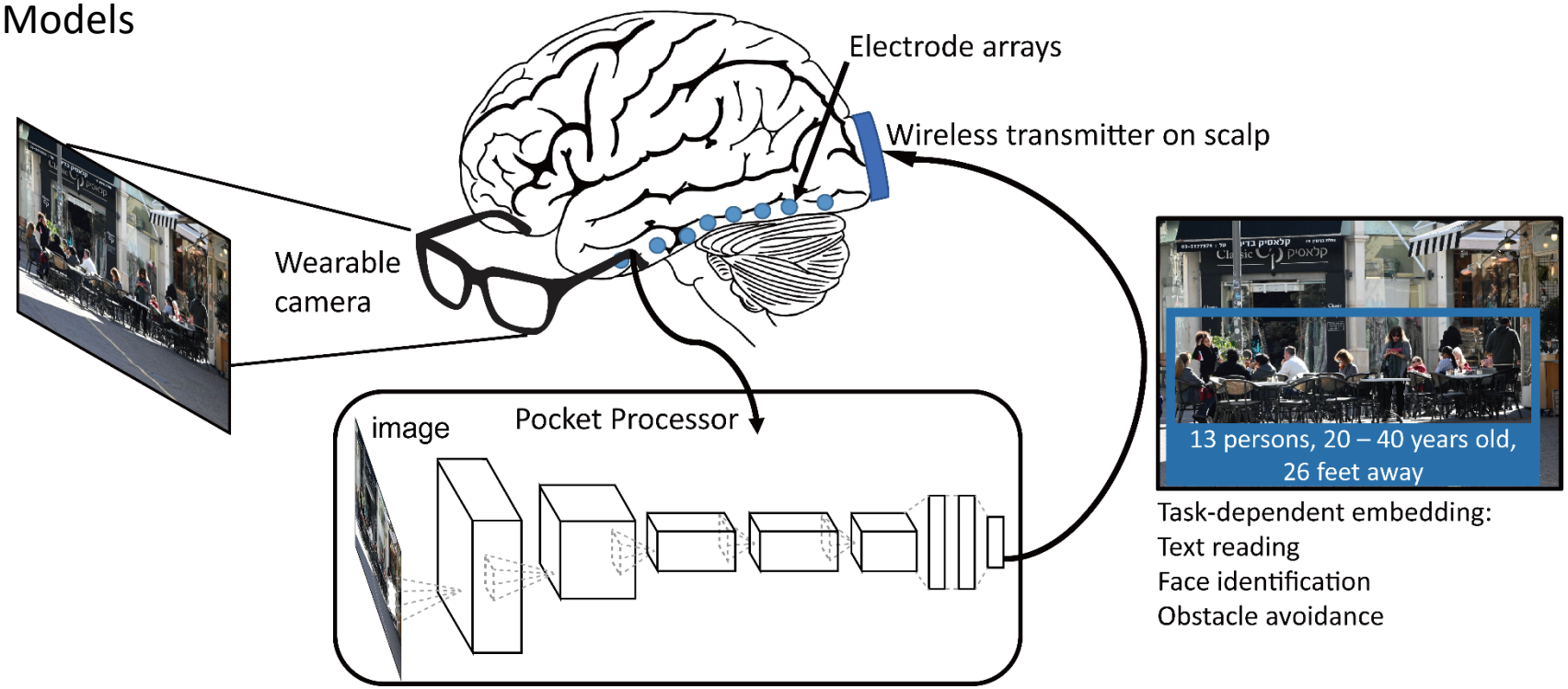
Towards Brain Computer Interface

Latent Codes In Generative Models



Latent Features In Recognition Models

Brain Machine Interface (BCI) for the visually impaired

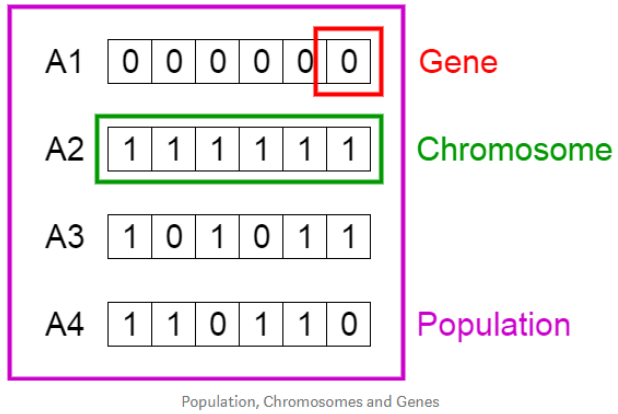


Task-dependent embedding:
Text reading
Face identification
Obstacle avoidance

Image credit: Xiao etal, unpublished figure, book chapter on AI and vision

Preliminaries on Genetic Algorithm

Natural selection process where the fittest individuals are selected for reproduction in order to produce offspring of the next generation. -> the fittest latent code z which drives neurons to fire as much as possible



Population, Chromosomes and Genes

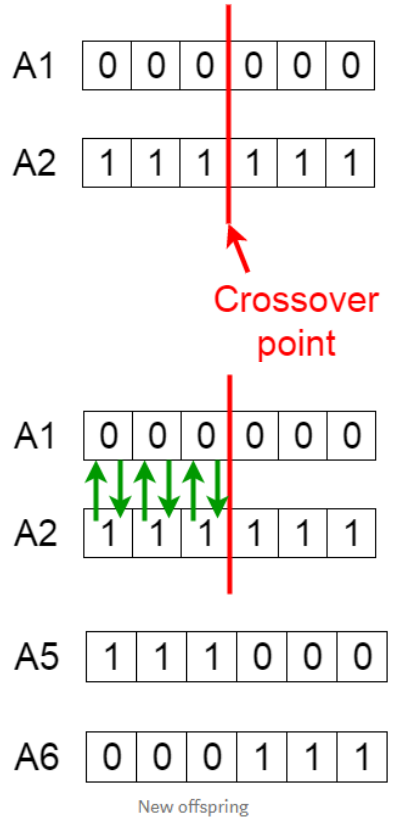
Fitness score
(Average Firing Rates)

A1	0 0 0 0 0 0	20
A2	1 1 1 1 1 1	15
A3	1 0 1 0 1 1	4
A4	1 1 0 1 1 0	2

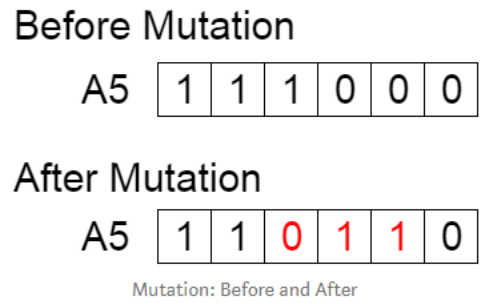
Selection

A1	0 0 0 0 0 0
A2	0 0 0 0 0 0

Fittest parents



New offspring



Mutation: Before and After

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

Image credits: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3#:~:text=A%20genetic%20algorithm%20is%20a,offspring%20of%20the%20next%20generation.

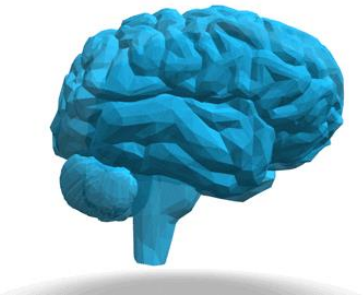
Preliminaries on fMRI data collection



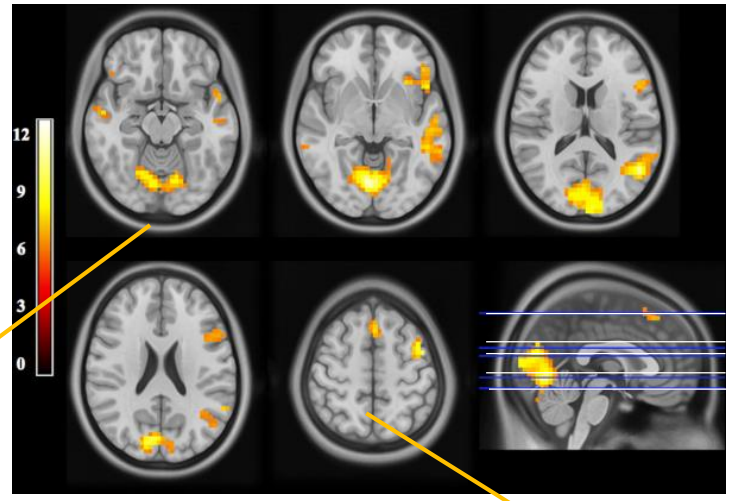
MRI machine

Image presentation

Controller



3D brain scans slice by slice



5 slices



Number of voxels per slice

Num of voxels - dimension vector for fMRI scans

Image credit: https://www.researchgate.net/figure/fMRI-maps-group-analysis-Statistical-maps-axial-slices-oriented-in-radiological_fig1_299516048; http://aucanlab.com/fmri/